SBIR Contract N00174-98-M-0412

Reconfigurable Computing

Final Technical Report
May 26 – November 26, 1998

Marcel van der Goot, Ph.D.
Scientist

Lee Fisher
Program Manager

Stephen Chau
Digital Design Engineer

Tanner Research, Inc.
2650 E. Foothill Blvd.
Pasadena, CA 91107
(626) 792-3000

1 Executive Summary

The field of micro-electronics is well known for its extremely high rate of change: components such as microprocessors, microcontrollers, and ASICs (Application-Specific Integrated Circuits), are replaced by new models every few years. This high rate of change is a mixed blessing for system designers. On the one hand, it allows them to address increasingly difficult problems with increasingly capable components; on the other hand, it means that any system built only a few years ago now contains old and obsolete parts. Thus system maintenance gradually becomes very difficult, if not impossible.

System design is a complex undertaking, as systems can contain from just a few to many thousands of components, and most modern systems contain software as well as hardware. Yet, any particular system is normally produced in vastly lower volume than its components, and more importantly, a system's life cycle is almost never well aligned with the life cycle of the components from which it is constructed. These factors combine to make it prohibitively expensive to update systems at the rate dictated by their component integrated circuits. As a result, it is typically the earliest innovators who find themselves holding large numbers of legacy systems, systems which can no longer be incrementally upgraded and will eventually fall into disrepair.

Reconfigurable hardware, primarily in the form of Field-Programmable Gate Arrays (FPGAs), provides an attractive solution to this problem. Advances in gate density, combined with the emerging intellectual property (IP) market of virtual components, make it possible to map a complete legacy system (typically, a board) to a small reconfigurable system. Since the same reconfigurable system can be used for many different applications, economies of scale are obtained. By using virtual components that match the original, but now obsolete, components, most redesign can be avoided. Moreover, since the reconfigurable hardware emulates the original system, legacy software can be readily reused.

A complete development system is required to enable effective use of FPGAs as replacements for legacy hardware. In addition to standard design tools, this system needs to include tools that integrate the development system with the reconfigurable hardware system. In Phase I we have implemented such a development system. It is in the form of a reconfigurable computer board containing multiple user-configurable FPGAs. The board is also equipped with a standard PCI interface that allows it to be used in a standard PC. The FPGAs can be configured using commonly available VHDL synthesis tools, and the PC interface greatly simplifies the development and testing of the configured design.

Furthermore, the reconfigurable computer has a modular design. In addition to one or two FPGAs, a module can incorporate special-purpose components such as memory or drivers which cannot be easily emulated by an FPGA. Moreover, once the configuration for the FPGAs has been developed and the interface to the PC is no longer needed, a module can be taken off and put on a special-purpose board fitting the form factor of the target legacy system. Since the development has already been performed and all the reconfigurable hardware is on the module, the special-purpose board needs to contain little more than wiring.

In Phase II, we plan to demonstrate the effectiveness of this approach by using the reconfigurable computer we have developed to prototype a Navy communication system. The intended target, selected in coordination with the sponsor, is a board in the Link-16 data commu-

nication system. This board is an evolution of the 1986-era Link-11. During our Phase I effort, we looked at the existing Link-11 design and used it as an indicator of the kind of component models that would be required to implement similar hardware. The Link-11 and Link-16 provide non-trivial examples to demonstrate both our ability to create legacy hardware replacements and to develop new hardware designs. Our long-term goals are to be able to map a large range of legacy systems to the reconfigurable computer in only a fraction of the time it would take to redesign these systems and to develop new solutions in equally rapid fashion. Due to the large number of legacy systems, such a system would have significant commercial applications, as well as fill an immediate need of the Navy and other branches of the Military.

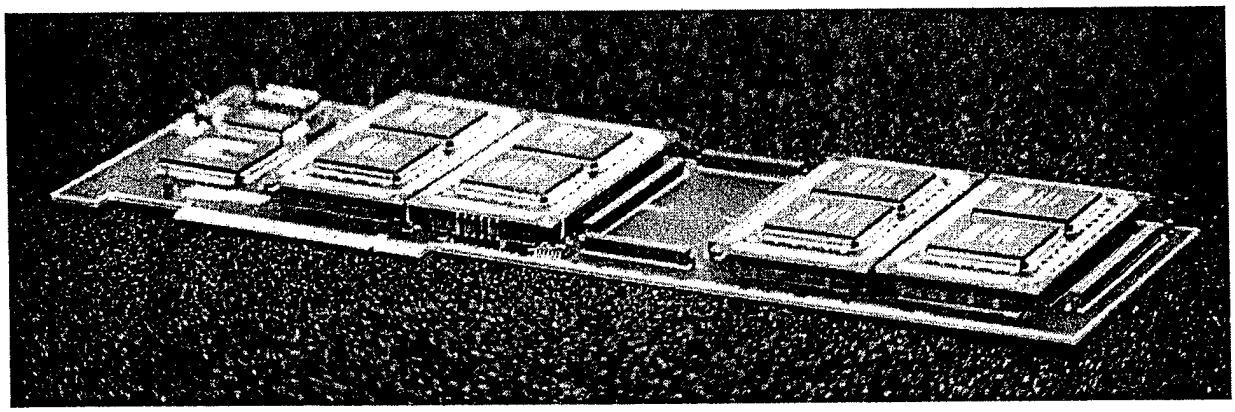


Figure 1: Photograph of the Tanner Research Reconfigurable Computer motherboard with four modules. Note that these modules are our earlier design, containing two FPGAs each; our new module will contain one FPGA plus half a MB of SRAM.

Figure 1 shows the existing Tanner Research Reconfigurable Computer motherboard with four dual-FPGA modules installed. This computer was originally developed under the Air Force Phase II SBIR contract F33615-94-C-1538, "Application Specific Electronic Design Synthesis: Neural Network Silicon Compiler". Originally designed to allow rapid development and implementation of Artificial Neural Networks, the solution is general enough that it can be readily extended to support implementation of other applications including legacy systems such as the Link-11. During this Phase I effort, we have extended and improved this earlier reconfigurable computer design, resulting in a new reconfigurable computer that is more powerful, more flexible, and more amenable to commercialization.

2 Identification and Significance of the Opportunity

Due to their desirable characteristics in terms of size, performance, and cost, complex integrated circuits, such as microprocessors, microcontrollers, and Application-Specific Integrated Circuits (ASICs), have become ubiquitous in electronics applications. However, the continuous rapid advancements in microelectronics technology cause such circuits to become obsolete very quickly. Once newer, faster parts are available, it soon becomes uneconomical to continue making the old parts in low volume. This poses serious problems for the users of systems built with such parts. While the obsolescence of the used parts does not immediately affect a working sys-

tem, it impacts the ability to repair the system as well as the ability of such systems to interact with newer designs. As a result, it often becomes necessary to design new systems – not because new functionality is needed, but because the parts needed to repair, duplicate, or make minor upgrades to the original system can no longer be obtained.

While it may be feasible to redesign certain mass-produced consumer electronics products every few years, such frequent redesign can be prohibitively expensive for larger and more critical systems. Among the factors contributing to the cost are:

- low volume (compared with consumer electronics);
- system complexity: large, complex systems (e.g., all the electronics aboard a ship) have many interacting sub-systems, making it hard to redesign only part of the system;
- system criticality: critical systems (e.g., hospital equipment, or aircraft navigation equipment) on which lives may depend, have very strict, and therefore costly, requirements regarding testing and verification;
- hardware-software interdependence: a redesign of the hardware may necessitate an even costlier redesign of the system's software.

Fortunately, the same technological advancements that cause the rapid aging of systems have also provided a possible solution in the form of configurable hardware. The most flexible form of configurable hardware is the Field-Programmable Gate Array (FPGA). While the gate density of an FPGA is several times lower than the density of a typical ASIC *in the same technology*, gate densities rapidly increase over time. As a result, by the time an ASIC or microprocessor becomes obsolete, FPGA technology typically has advanced enough to be able to emulate the original part. Furthermore, if the gate count of a single FPGA is insufficient, a large design can be distributed over multiple FPGAs. Emulation need not be restricted to a single part either: since in most cases more than one component of a board will be obsolete, it often makes sense to emulate the whole board directly in the FPGA.

In Phase I we started with a reconfigurable computer that was developed on our Air Force Phase II SBIR contract F33615-94-C-1538, "Application Specific Electronic Design Synthesis: Neural Network Silicon Compiler." From this earlier design, we have developed a new more flexible, modular, and reconfigurable computer that provides a convenient development system for configurable hardware. The reconfigurable computer consists of a motherboard with a PCI interface, and from 1 to 5 FPGA-based modules. The PCI interface connects the reconfigurable computer board to a host PC; this interface is used both to load configuration data into the FPGAs and to enable the host PC to communicate directly with each of the modules. Figure 2 shows the baseline hardware installed in a PC. Section 3.1.3 provides a more in-depth description of the improved reconfigurable computer architecture and hardware we have developed under this contract.

The modular design that we have implemented adds an extra dimension to the configurability of the computer: not only is the function of the FPGAs completely user-configurable, but new modules containing special-purpose hardware can easily be incorporated into the computer. The result is a system that is readily tailored to a specific application area. The existing module set includes one module-type with two FPGAs and another module-type with one FPGA and some RAM. Should a new application require other components, such as A/D converters or drivers for different voltages, a special-purpose module can be readily implemented to provide them.

The modularity of the reconfigurable computer is particularly useful when replacing legacy hardware. For example, during the development phase of a project, a designer might be trying to duplicate the functionality of a legacy hardware/software system. In this situation, the PC interface and host environment is used extensively. Once the new system is working, however, the PC interface and associated hardware will most likely no longer be needed. Since all of the desired functionality of the new system is provided by the module or modules on the reconfigurable computer and their application-specific configuration, they can be removed and put on a separate board which fits the form factor of the legacy system. This new board needs to contain little more than wiring and sockets for the modules needed. The modules provide the FPGA-based circuit configuration and any application-specific components or specialized I/O interfaces needed by the legacy application.

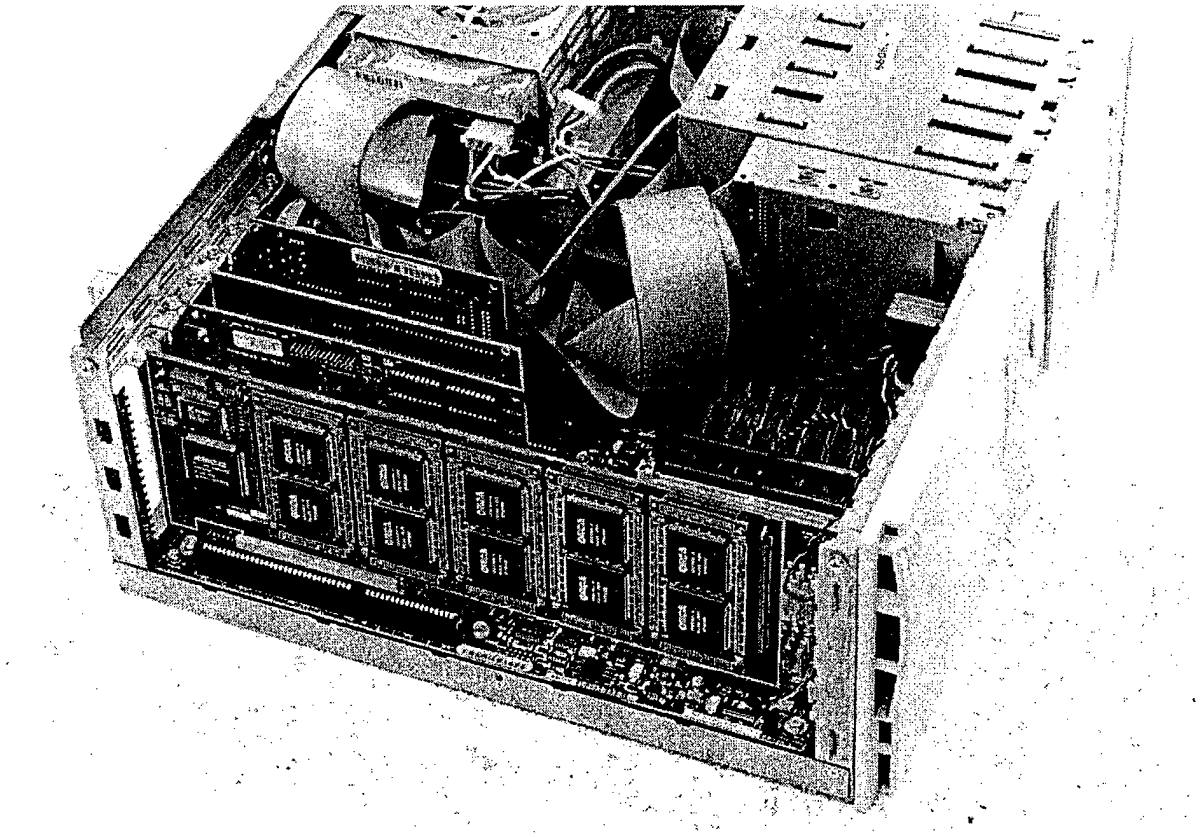


Figure 2: Photograph of the Tanner Research Reconfigurable Computer mounted in a PC.

This modular design method makes it straightforward to move from our development system to a production board compatible with the legacy environment, and it simplifies the use of the configurable hardware as part of a larger legacy system. Furthermore, because the PC interface and other supporting circuitry in our development system are not included in the production board, production costs can be minimized.

It should be pointed out that replacement of legacy hardware is but one of the many applications of the Reconfigurable Computer. In particular, it provides an excellent implementation method for new systems: configurable hardware is considerably cheaper than custom ASICs. In

addition, it allows for easy improvement in both functionality and performance as new technologies become available or application requirements change. The latter is of particular interest when a system is difficult to reach, an example being the hardware within an orbiting satellite. Reconfigurability is also important in areas such as communication, where protocols and standards change frequently. The use of configurable hardware for new systems reduces the likelihood of creating future legacy hardware.

3 Phase I Technical Work

The primary goals for the first six months of our Phase I work have been to:

- Determine the technical requirements for a reconfigurable computer suitable for developing legacy hardware requirements
- Design the prototype reconfigurable computer architecture
- Design the prototype reconfigurable computer hardware
- Submit the prototype hardware for fabrication

The remaining goals of the Phase I (the Option) effort are to:

- Develop the interface software necessary to support use of the reconfigurable computer in a PC host environment, and
- Assemble, test, and deliver a completed prototype.

The methods we used and the results we achieved are described in the following sections. The design we have produced is documented in Appendices A through E.

3.1 Determination of Technical Requirements

The initial goal of the first six months of Phase I was to determine the technical requirements for a reconfigurable computer suitable for developing legacy hardware replacements. To achieve this goal, we began with the analysis of an example legacy system.

3.1.1 The Link-11 System

In cooperation with our sponsor, we selected the Link-11 Interface Converter CV-3878/SWG (LIC) as our legacy hardware example. The LIC is capable of monitoring two non-simultaneous channels of 26/29 bit parallel data that is formatted according to MIL-STD-1397, Type A, Category I, with electrical characteristics defined by MIL-STD-188-203-1. Each 26- or 29-bit parallel data word is reformatted into four 8-bit parallel data words, which the LIC converts to asynchronous serial format (MIL-STD-188C) for the Tomahawk Weapon System. The core components of the Link-11 Interface Converter are three identical printed circuit boards, the *TRANSMIT* PWB (printed wiring board), the *RECEIVE* PWB, and the *SPARE* PWB; this is the legacy board that we will map to the reconfigurable computer.

Providing LIC capability using the reconfigurable computer (one of the goals of Phase II) will provide an excellent demonstration of the intended methodology for legacy system replacement. LIC boards currently in use are deteriorating and will soon need to be replaced, but their design is

twelve years old and all major components are now obsolete. The sponsor is in need of a low cost approach to building more Link-11 data converters. Among other integrated circuits, the LIC board contains an 8085 microprocessor, with associated software stored in an EPROM. Mapping the microprocessor to the board will demonstrate that the approach applies to non-trivial hardware. Furthermore, since the mapped board will emulate the original processor, the software need not be modified at all, demonstrating another strength of the approach.

The sponsor anticipates converting to a new protocol, Link-16, in the near future. Link-16 can be seen as an improvement of Link-11: while a Link-16 system will use the same interface, it will offer improved performance in areas such as throughput and jam resistance. Using reconfigurable hardware to implement the Link-16 will allow deployment of new Link-16-capable systems that are also backwards-compatible with the legacy Link-11. Thus, we will focus our Phase II efforts on prototyping Link-16 functionality with Link-11 capability.

3.1.2 Virtual Components

An important aspect of determining the technical requirements for the Tanner Research Reconfigurable Computer was an analysis of the feasibility of mapping the chosen legacy system onto the reconfigurable computer using virtual components. Virtual components are pieces of code by which existing hardware component functionality can be "replicated" within an FPGA. Hence, they have both hardware and software aspects. The virtual components are typically described in VHDL or as a gate-level netlist.

For this project we investigated two classes of virtual components. The first class consist of components used to access the resources of the board. A good example is the FPGA plus RAM module. A useful virtual component might make it easy for the FPGA to access the memory by providing merely an address, without the need to consider the actual signaling sequence involved. Another virtual component will be used in conjunction with the device driver on the host, to provide direct host access to the module's RAM. This latter virtual component will be developed in conjunction with the corresponding software library routine during the Phase I Option.

The second class of virtual components are those which mimic existing hardware components. This class of virtual components is crucial to our approach to replacing legacy hardware, since substantial redesign of a legacy system can only be avoided when the virtual building blocks used closely match the original components. Because the goal of this project is to explore the use of such virtual components to replace legacy hardware, not the development of such components per se, we have explored the commercial availability of such parts. (Note that the actual use of these components is planned for Phase II, not for the current Phase I.)

The components used by the Link-11 board (LIC) are:

- 8085A: an 8-bit microprocessor;
- 8251A: a programmable communication interface ("UART");
- 8255A: a programmable peripheral interface;
- 8254: a programmable interval timer;
- various TTL components, including demultiplexers, latches and flip-flops, NANDs, and inverters.

We have identified a vendor, CAST Inc., which sells virtual components matching all of the above, except for the 8085 (i.e., the microprocessor). Some of these components are also available from other vendors. Prices for the 82xx components are typically from $5,000 to $10,000 each (depending on the vendor and on the type of license); a library containing the simple TTL components is substantially cheaper. The typical size of an 82xx virtual component is around 6,000 gates, which would easily fit in a single FPGA.

While there exist synthesizable models of various 8-bit microprocessors and microcontrollers (e.g., many vendors sell 8051 models), we have been unable to find an 8085 core. There is a free model for a "GL85," which is supposed to be instruction-compatible, but not pin-compatible; also, it is unclear whether this model, which is primarily intended as an example, is sufficiently robust or complete for our purposes. However, considering the availability of similar processor cores, both commercial and public-domain, we feel confident that obtaining an 8085 core will not be a stumbling block for this project. We will decide whether to design it ourselves during the Phase II, or whether to hire an outside firm to provide one (we have already received at least one offer). Typical existing processor cores of similar complexity cost from $20,000 to $40,000; a custom-made model would probably cost about $20,000 more. Existing models contain from 10,000 to 20,000 gates, which again would pose no problem for a single FPGA.

In addition to the above-mentioned components, the Link-11 board contains an 8KB static RAM, an 8KB EPROM, and various input and output drivers. The RAM and EPROM can be modeled by the module's memory, with the EPROM contents initialized by the host PC. We do not expect that input drivers will be needed, as the FPGA has its own built-in input drivers. The FPGA also has output drivers, but at this moment it is unknown whether these match the electrical requirements of the Link-11 board (we have not yet received those requirements). In either case, it will not be a problem to add suitable drivers if necessary.

3.1.3 Mapping Virtual Components onto FPGAs

Developing a configuration for the FPGAs normally consists of two distinct phases:

1. Creating a high-level behavioral description in VHDL, and synthesizing this description to a gate-level implementation.
2. Mapping the gate-level circuit to the FPGA primitive cells, including routing the signals.

In theory, step 1 can be bypassed by directly designing a gate-level implementation using a schematic editor; in practice, that approach only works for small designs, and is therefore not suitable for our intended application. One can also avoid step 1 by buying gate-level virtual components (see Section 3.1.2), but that merely means that step 1 was performed by someone else.

In our Phase II effort, we will use commercially available software to perform both synthesis steps. In particular, we will use "FPGA Express" by Synopsis to perform the high-level synthesis (i.e., step 1); this software can be targeted for a variety of FPGAs. For step 2 we will use Lucent's "bitgen" program, which is specifically intended to perform mapping for Lucent FPGAs. While Synopsis is probably the most widely used vendor of VHDL synthesis software, this does not prevent other users of the reconfigurable computer board from using different synthesis tools. On the other hand, it seems likely that whatever high-level synthesis tool is used, the low-level mapping tool will always be Lucent's "bitgen."

In addition to the synthesis tools, a VHDL simulator is essential for any non-trivial design. Again, a variety of simulators can be used; in Phase II, we will use the "V-System" simulator from Model Technologies.

3.2 Architecture and Hardware Design

Based in part on our analysis of the Link-11 system, we have adapted our earlier reconfigurable computer board design to the requirements imposed by this analysis. This earlier design consisted of a motherboard and up to five dual-FPGA modules. The redesign we accomplished during this Phase I activity included minor revisions to the motherboard intended to improve producibility, reliability, and maintainability, as well as the design of a new module, one consisting of an FPGA plus RAM. Figure 3 presents the block diagram of the improved design.

3.2.1 The Tanner Research Reconfigurable Computer Motherboard

The motherboard contains the PCI interface that connects the reconfigurable computer to the PC. Physically, the interface consists of the connector, a PCI interface chip, custom logic, and a bus connecting the modules. The PCI interface chip and custom logic together translate PCI bus communications to selection signals and data transfers to the modules. From the device writer's point of view, the interface consists of selection and data registers. The custom logic is in the form of a Complex Programmable Logic Device (CPLD), a configurable chip similar to an FPGA. The CPLD can be configured on the board, through a standard JTAG interface, and retains its configuration even when the power is switched off. The CPLD is not intended to be user-programmable, but its programmability facilitates future upgrades to the hardware. The PCI interface chip is somewhat configurable through setup values stored in a small non-volatile memory chip.

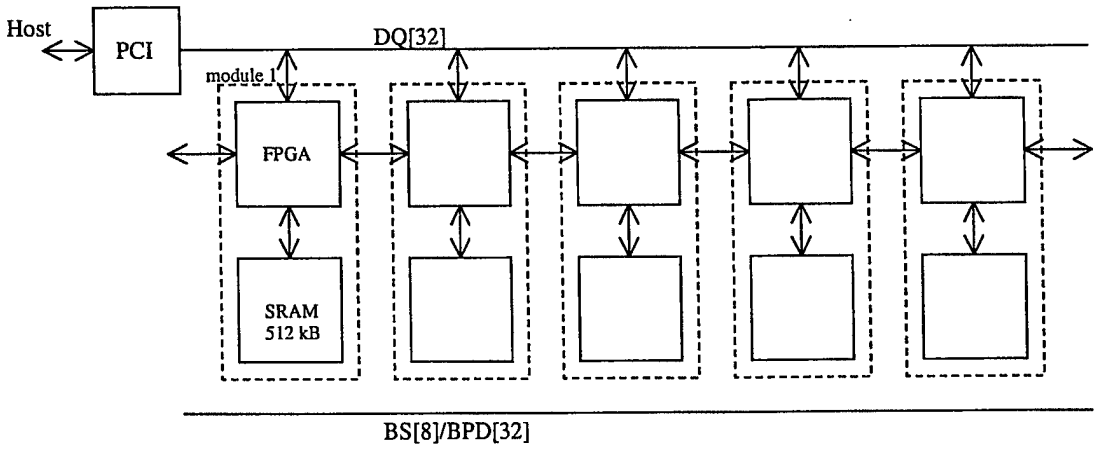


Figure 3: Reconfigurable computer, busses and data wires. For emulation of the Link-11 board only one module will be needed. The BS and BPD busses are for use with dual-FPGA modules.

In most cases, all logic is synchronized with the PCI clock. However, the board also has an optional programmable oscillator which can generate a large range of frequencies to be used as a

clock signal for the FPGAs. This is of particular interest for legacy hardware replacement, as it may be required that the replacement operates at the same frequency as the original system. In cases where only one particular frequency is needed, the programmable oscillator can be replaced with a less expensive fixed-frequency oscillator.

The motherboard contains sockets for up to five modules ("daughtercards"), and includes all necessary wiring. Data wires come in the form of busses connecting all modules as well as in the form of channels connecting neighboring modules (all are at least 32 bits wide). In addition to the data wires there are module selection signals, clock signals, and 3.3V, 5V, and +/−12V power lines. During this Phase I effort, a number of minor improvements were made to the motherboard, including a change of sockets and improved power routing. The layout of the motherboard is presented in Appendix A. Signal layers, power and ground layers and assembly drawings are included. The schematics of the motherboard are in Appendix D.

3.2.2 The Tanner Research Reconfigurable Computer FPGA plus RAM Module

In earlier designs we used modules consisting of two FPGAs, (the layout of the Dual-FPGA module is presented in Appendix B and the schematics are in Appendix E) but for this project we have designed a module with a single FPGA plus RAM. This module matches the requirements of the Link-11 example better than the dual-FPGA module: while an FPGA can store some data, it would not be efficient to use an FPGA to emulate the 16 KB of memory present on the Link-11 board. The FPGA plus RAM board contains 128 KB of 32-bit wide, static memory (SRAM). Compared with our earlier design, the FPGA has been updated to the Lucent ORCA 3C80, and can now support up to 80,000 gates. The new FPGA is logically similar to, but not quite pin-compatible with, the earlier part. As a demonstration of the power of the modular design, the motherboard did not need any changes to accommodate the new module, even with the newer FPGA.

The memory on the module is directly connected to the FPGA, not to the PCI bus. However, it is straightforward to configure the FPGA in such a way that the module's memory is directly mapped to the host computer's address space. Hence, a user can read and write the memory directly, greatly simplifying development and debugging.

Based on the complexity of the Link-11 board and the capacity of the FPGA, one module should provide ample resources to emulate the whole board. However, if there is a need to emulate more complex designs, one or more additional modules can easily be added.

The layout of the FPGA plus RAM module is presented in Appendix C while the schematics are in Appendix F.

3.3 Hardware Fabrication

Based on our success with the prior reconfigurable computer, Tanner Research's Electronic Products Division (EPD) has committed to pursuing commercial applications of the computer. To that end, EPD completed the physical design (layout) of the redesigned boards and initiated their fabrication. The new motherboards have already been submitted for fabrication and the modules will be submitted shortly. The support of EPD has allowed us to create a substantially better hardware product than our Phase I funding alone would have permitted, since they were able to direct particular attention to issues important in the commercial success of the product. Reliability was an issue to which they paid particular attention, for example. In addition, because of the support we received from EPD, we were able to give greater emphasis during this Phase of

the effort to the investigation of the virtual components needed to implement the Link-11 legacy system in the reconfigurable computer.

3.4 Remaining Phase I effort (the Option)

The remaining effort on Phase I (the Option) will focus on the development of the interface software necessary to support the use of the reconfigurable computer in a PC host environment in addition to the delivery of an assembled and tested prototype.

The software used with the reconfigurable computer falls into two quite different classes: the software used to develop the FPGA configuration, and the interface software used to communicate with the reconfigurable computer board. While we will use commercially available software to design and create FPGA configurations in our Phase II effort (see Section 3.1.3), we have found that a simple interface allowing the transfer of data between the host PC and the FPGAs is an essential tool needed for the development process. Since there is no commercially available interface software available for our reconfigurable computer, we will develop the necessary software in the Phase I Option. We will then use that software in Phase II.

3.4.1 Interface Software

Interface software has already been written for our earlier board design. This software is in the form of a stand-alone program. It includes a graphical user interface (GUI) which allows access to various resources of the reconfigurable computer. While this software is useful for developing and testing the board itself, it is less convenient as a basis for developing applications on the PC that use the board. Additionally, the existing software does not support all of the functionality of the board. Consequently, in the Option portion of Phase I we plan to rewrite this software completely. We will write it in the form of a library of user-callable routines rather than as a stand-alone program. Such a library is often called a device driver.

We will implement several categories of interface routines, where their categorization is based on their function as well as on the level of abstraction they provide. The first category is that of configuration functions. Functions in this category provide the following capabilities:

- PCI setup;
- Board detection;
- FPGA configuration;
- FPGA contents read-back;
- Oscillator configuration.

These are all rather low-level functions. PCI setup and board detection are normally done automatically, and these functions are present primarily for completeness. FPGA configuration is the most used function in this category, especially during development, when the configuration is changed often. Configuration consist of down-loading the binaries produced by the synthesis tools to the board's FPGAs. Read-back of the FPGA contents provides a low-level debugging method; it is not used very often, because most debugging is more easily done using a VHDL simulator. The oscillator only needs to be configured for applications which cannot run at the PCI clock frequency.

The second category contains the low-level data communication functions:

- Module selection;
- Register read and write;
- Memory-mapped read and write;
- FIFO-based communication.

These functions directly access the PCI interface to communicate with the board's modules. Module selection is used to set the target for following reads and writes (multiple modules can be selected simultaneously to provide data broadcasts). Register reads and writes communicate with the FPGAs through a single 32-bit PCI register, whereas memory-mapped reads and writes communicate by sharing a region of memory. For applications which need high throughput, the PCI interface provides a FIFO protocol.

Additional, higher-level functions will be implemented as necessary to facilitate the testing of the delivered prototype hardware in Phase I. An example of such a function might be a routine that reads from and writes to the RAM of an FPGA plus RAM module. Creation of the Link-11 emulation in Phase II may drive the addition of new interface software (device driver) capability as well. This capability will be added in Phase II as needed.

3.4.2 Assembly, Test, and Delivery of the Prototype

During the remaining portion of Phase I (the Option), we will complete the assembly and test of a prototype reconfigurable computer.

4 Future Research and Development

The goals of this project have been to demonstrate the use of reconfigurable hardware as replacement for legacy systems, and to develop the hardware and software needed to effectively create such replacements. The Phase I work described in this Final Report has focused on our investigation into the means by which legacy hardware can be re-implemented in a reconfigurable computer and the development of a new reconfigurable computer that will support the re-implementation of an example legacy system, the Link-11. Development of a prototype Link-16 system incorporating backwards compatibility to the Link-11 will be the major task of the Phase II effort.

The remaining part of Phase I (the Option), has two major tasks: developing the interface software, and fabrication and testing of the hardware. As described in Section 3.4.1, the interface software development will consist of writing several levels of library routines. The fabrication of the hardware has already been initiated by EPD, leaving ample time to test the assembled hardware and develop the interface software. The tested hardware together with the interface software will form a complete reconfigurable computer system, which we will deliver at the end of the three month Option period.

Having a complete reconfigurable computer by the end of Phase I will allow us to concentrate in Phase II on the task of prototyping the Link-16. A major technical issue during this prototyping effort will be the resolution of timing requirements. Even given accurate VHDL models of the different components, the actual timing is often hard to predict, due to the uncertainties of high-level synthesis and, in particular, the placement and routing of gates in the FPGA.

Once the Link-16 board has been successfully prototyped using the reconfigurable computer, we will explore the issues involved in porting the configured module to a new motherboard that does not have the PCI interface, but instead can directly be inserted into the Navy Link-16 communications system.

Our long-term goal is to build a complete development system, based on the reconfigurable computer, which will provide an economical and straightforward solution for the replacement of a large variety of legacy systems and the prototyping of new systems. Such a system would be of immediate benefit to the Navy and other branches of the Armed Forces, and would have many commercial applications as well. In addition, while the replacement of legacy systems places its own unique demands on the reconfigurable computer environment, reconfigurable computers have many other application areas. One particular area of interest is the design of new systems directly using reconfigurable hardware, both to allow easy upgrades and to avoid a future legacy problem. Tanner Research is developing a commercial reconfigurable computer suitable for such applications.

5 Appendix A: Motherboard Wiring

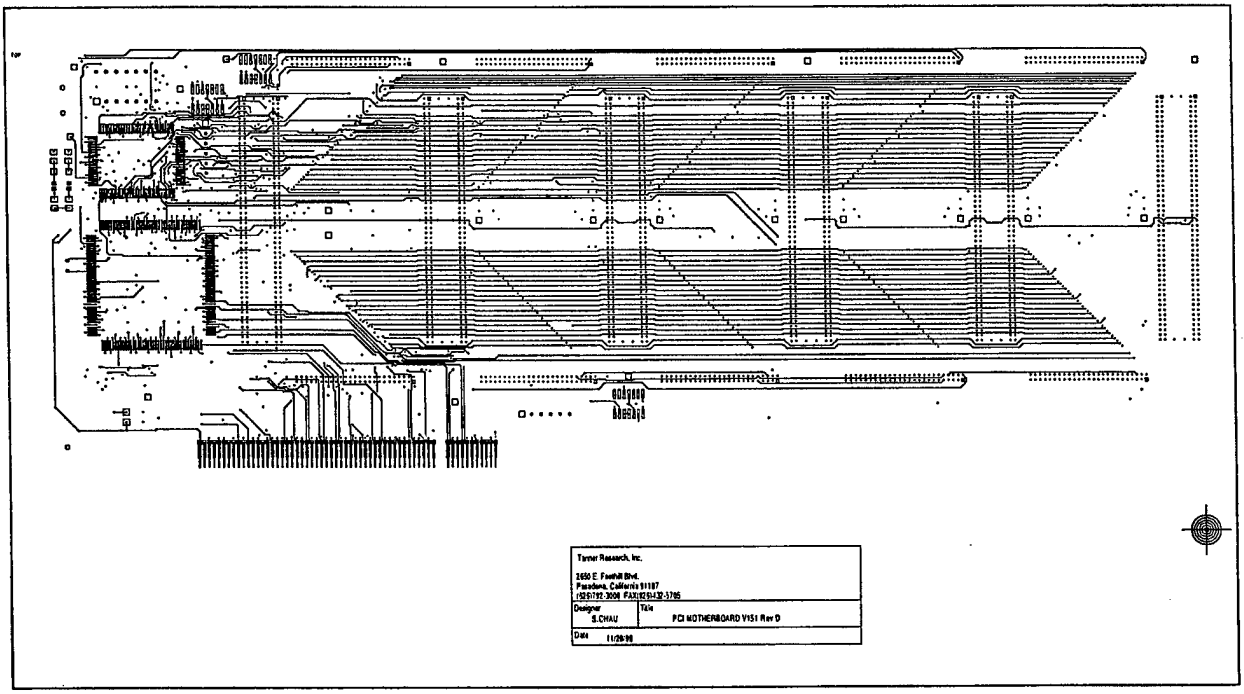


Figure 4: Motherboard Top Layer Trace Layout (15381top)

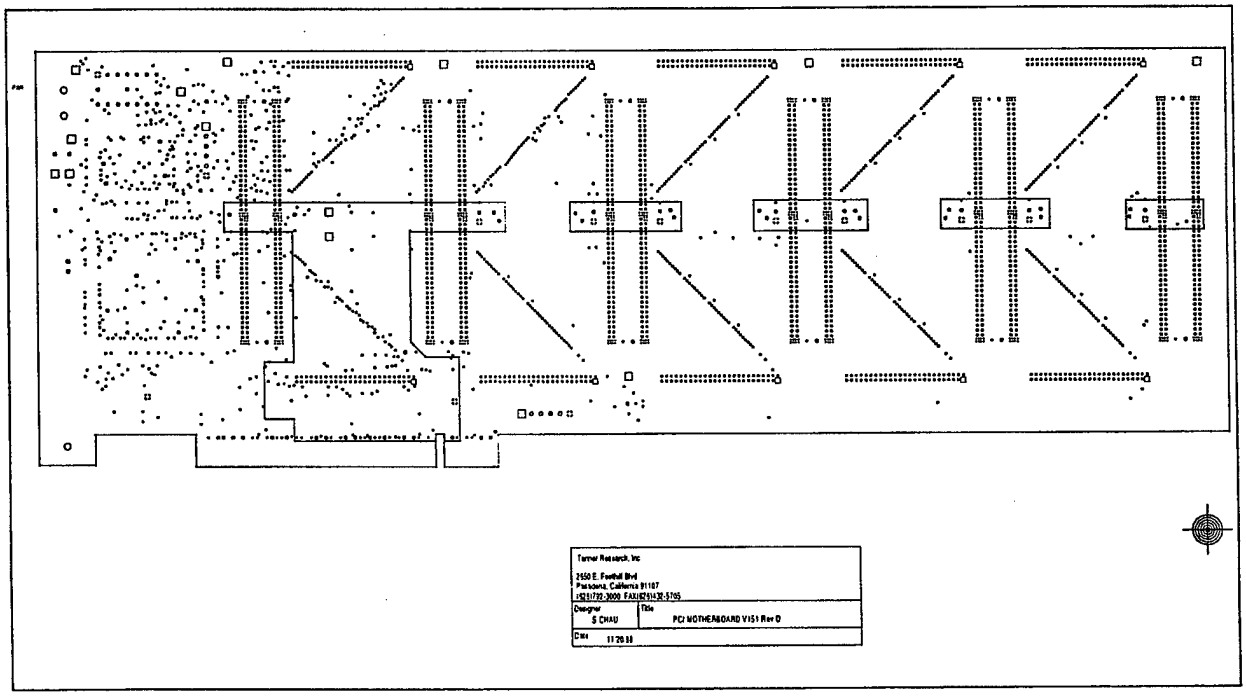


Figure 5: Motherboard Split Power Plane Layout (15381pwr)

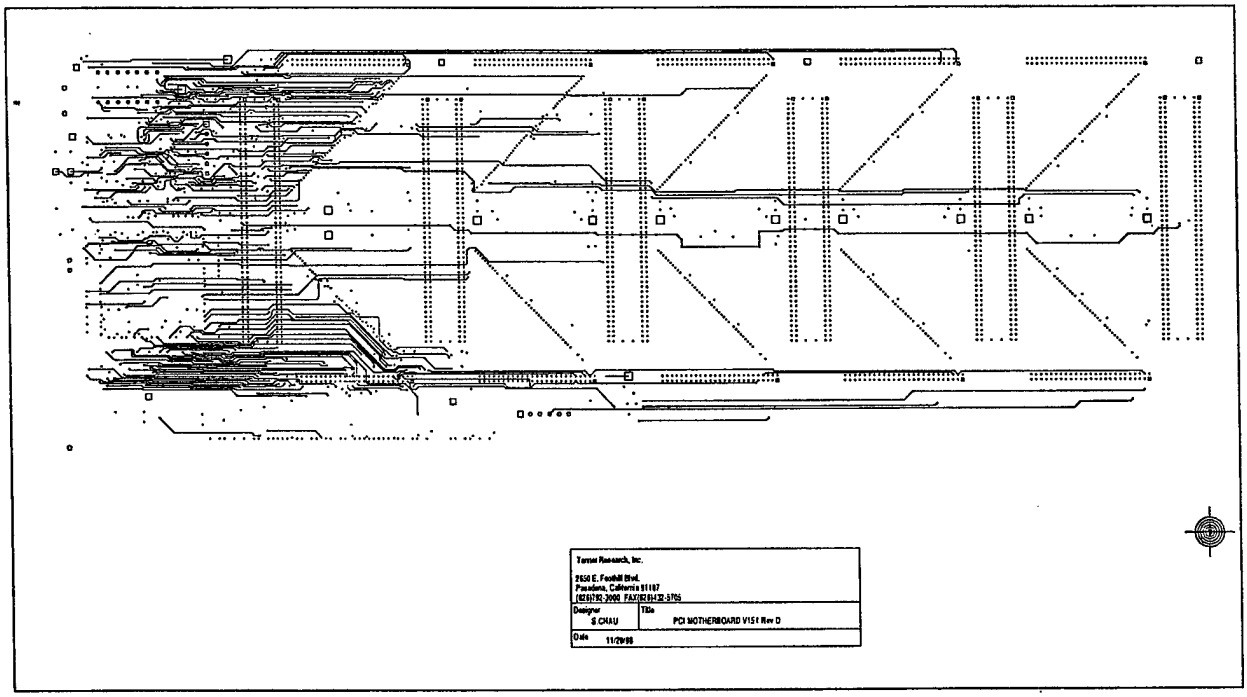
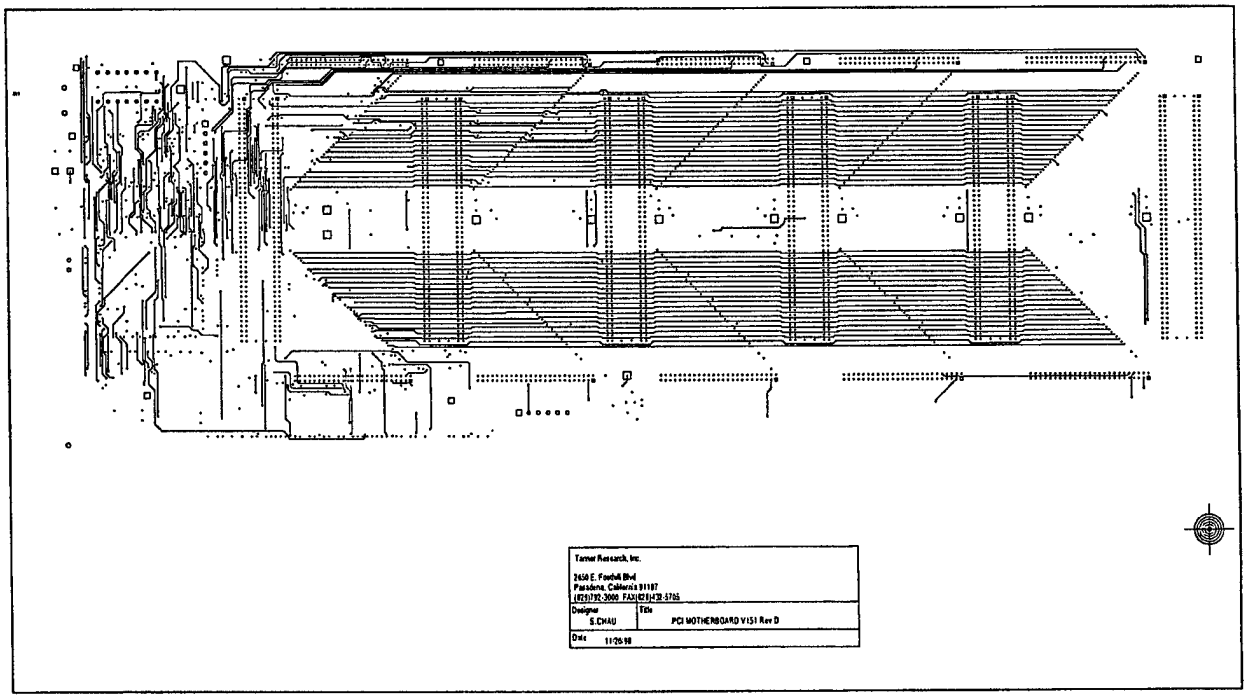


Figure 6: Motherboard 2nd Internal Signal Trace Layout (15381in2)



1 Figure 7: Motherboard 1st Internal Signal Trace Layout (5381in1)

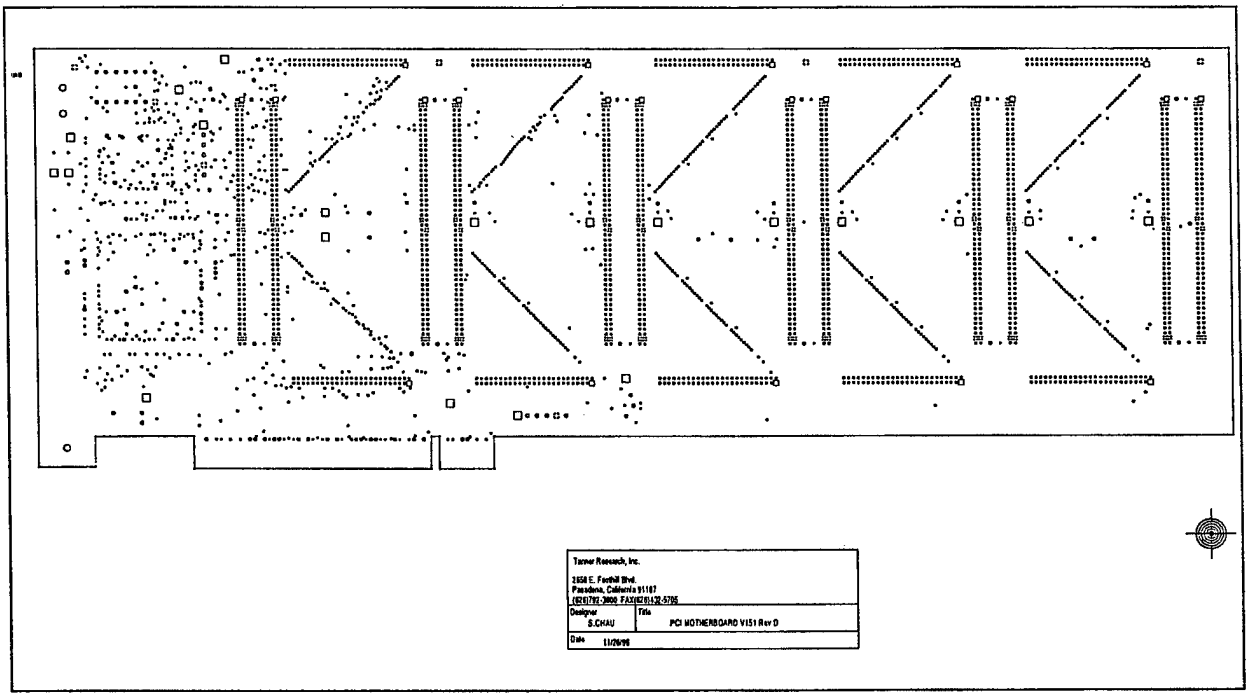


Figure 8: Motherboard Common Ground Plane Layout (15381gnd)

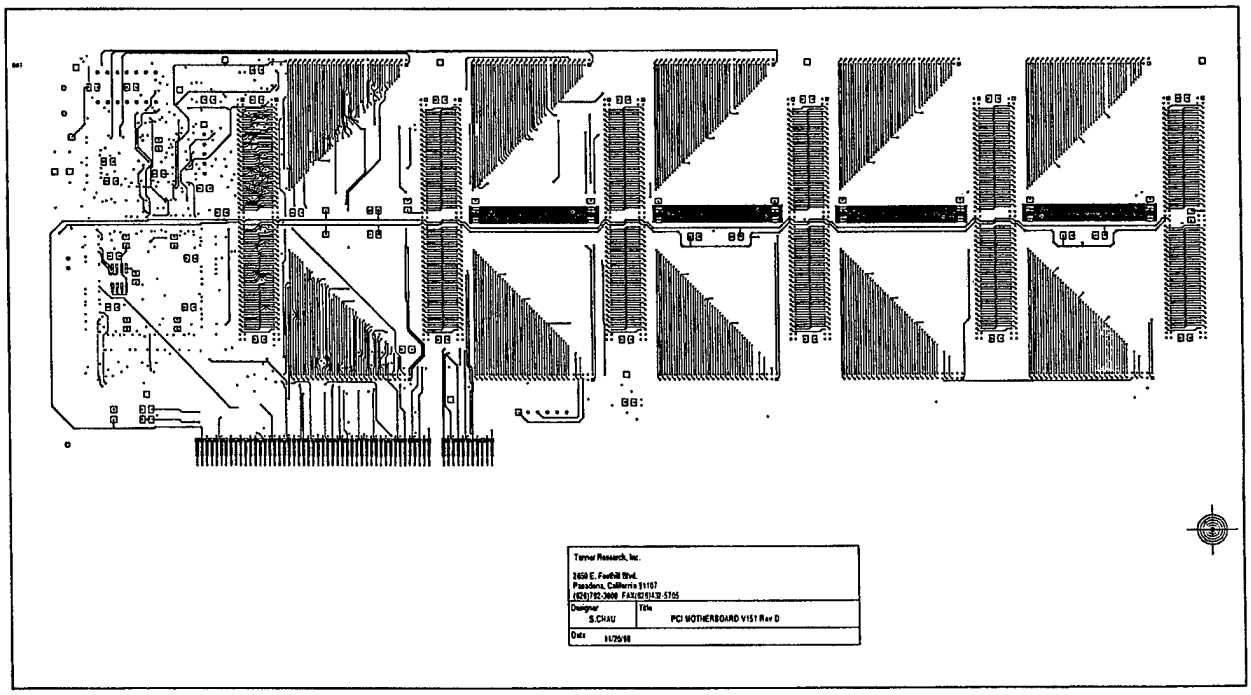


Figure 9: Motherboard Bottom Layer Trace Layout (15381bot)

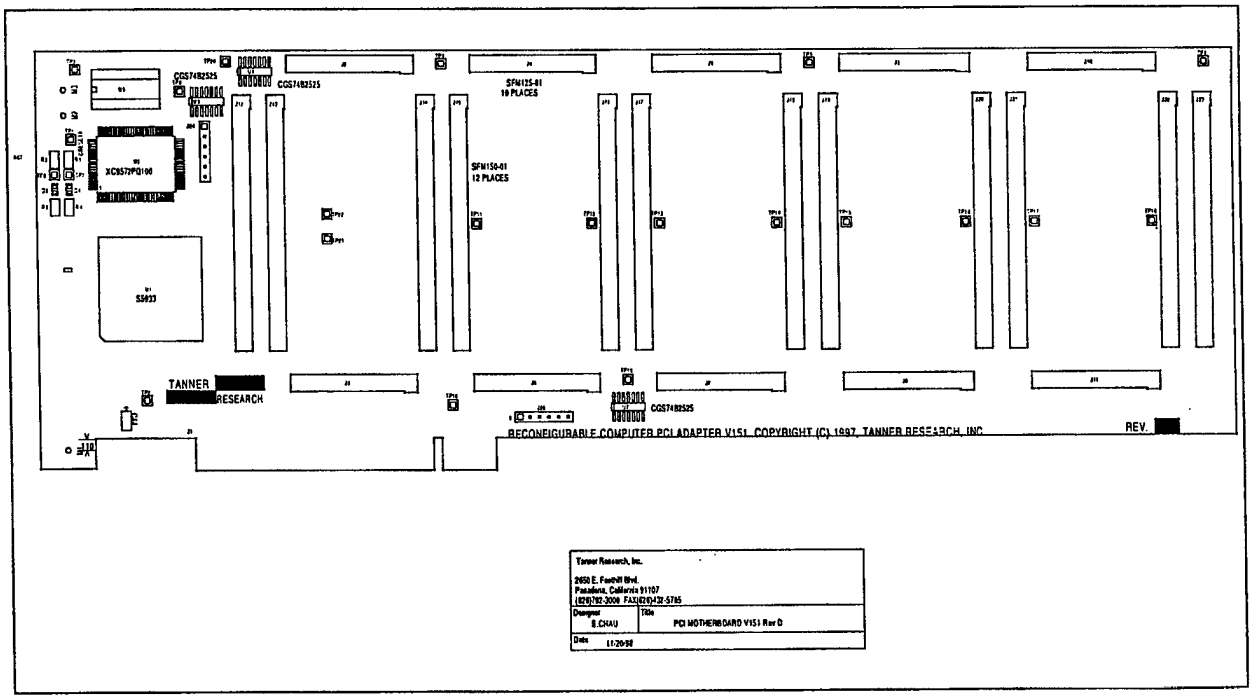


Figure 10: Motherboard Top Layer Assembly Drawing (15381ast)

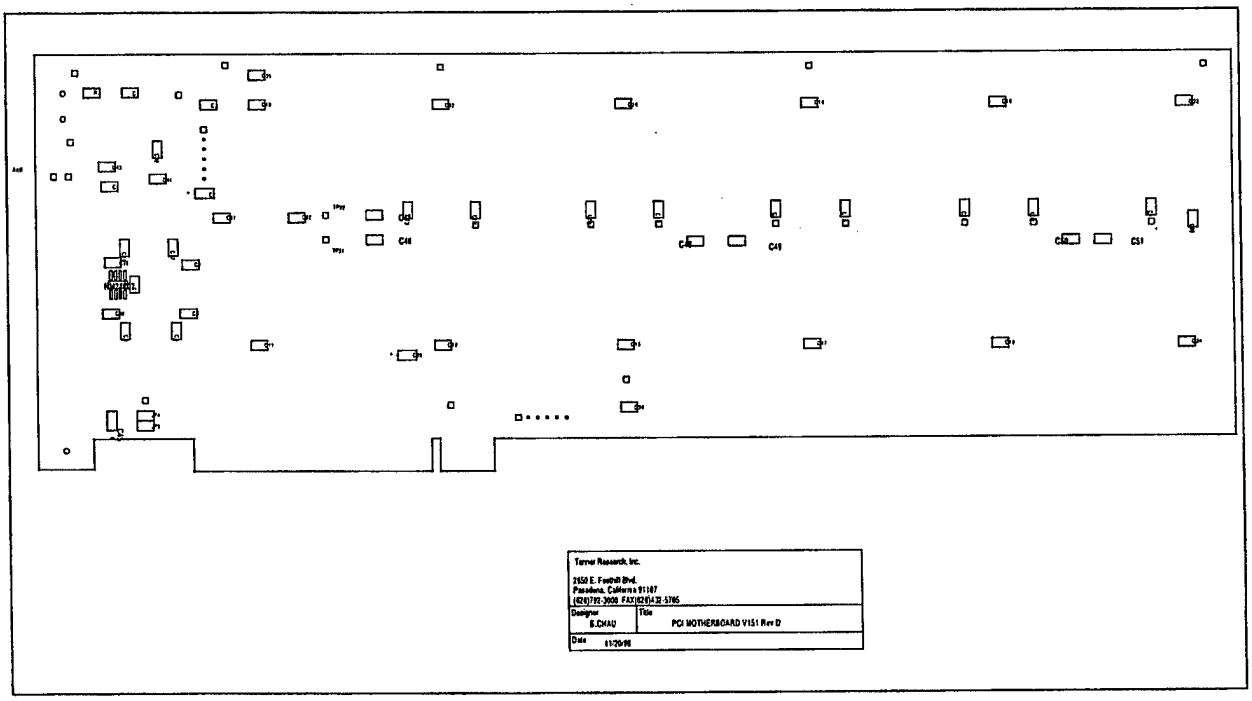
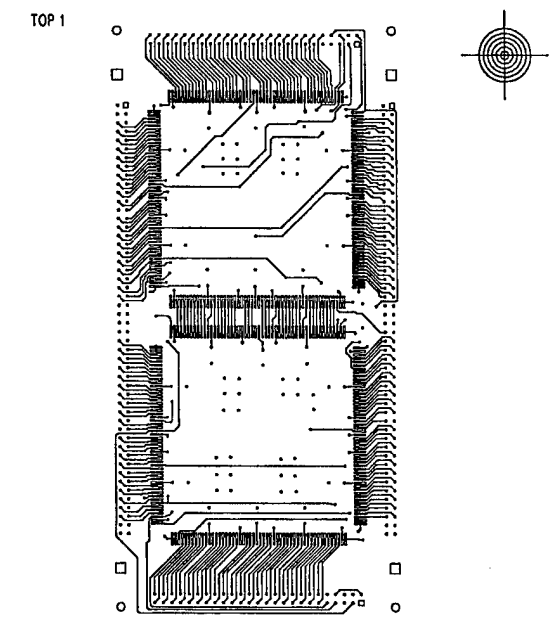


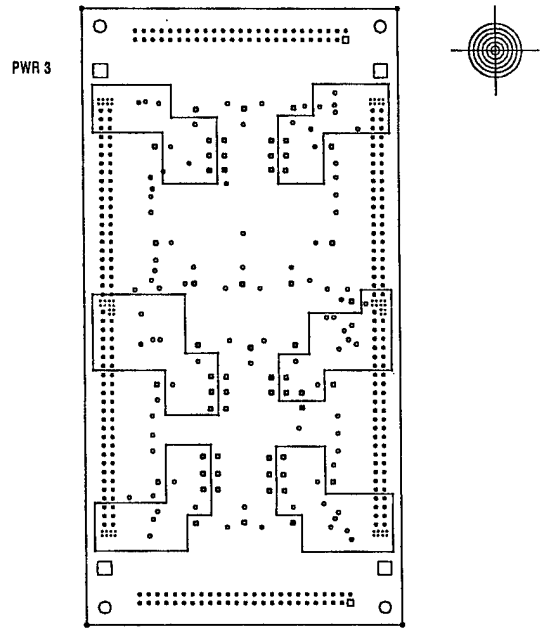
Figure 11: Motherboard Top Layer Assembly Drawing (15381asb)

6 Appendix B: Dual FPGA Module Wiring



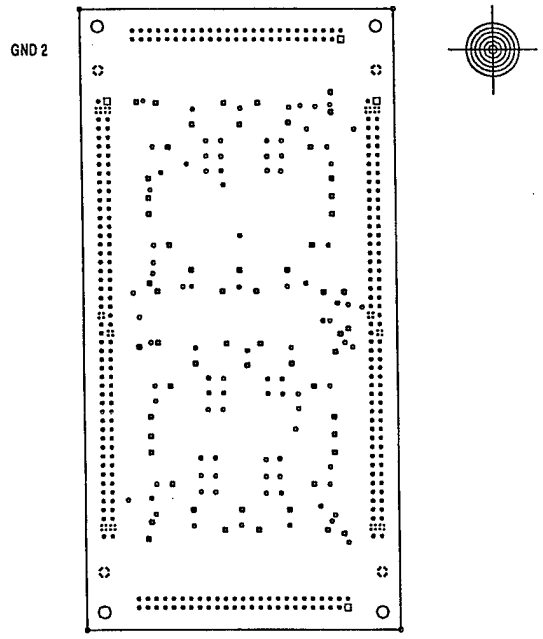
Tanner Research, Inc. 2650 E. Foothill Blvd. Pasadena, California 91107 (626)792-3000 FAX(626)432-5705	
Designer S.CHAU	Title DUAL FPGA DAUGHTER BOARD V100 Rev. A
Date 11/23/98	

Figure 12: Dual FPGA Top Layer Trace Layout (fpgatop)



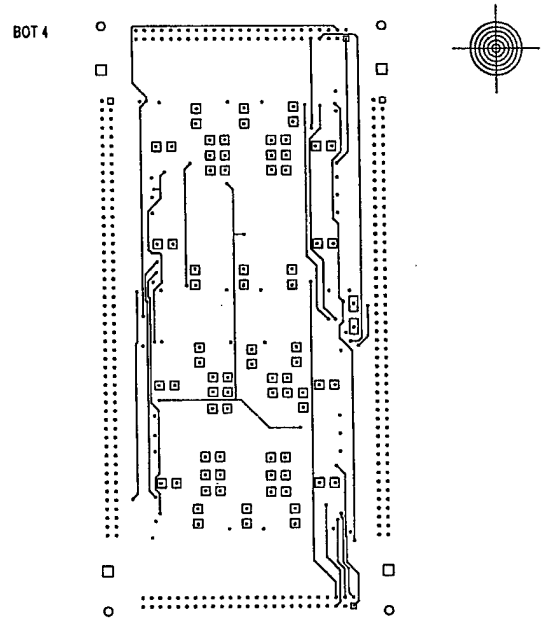
Tanner Research, Inc. 2650 E. Foothill Blvd. Pasadena, California 91107 (626)792-3000 FAX(626)432-5705	
Designer S.CHAU	Title DUAL FPGA DAUGHTER BOARD V100 Rev. A
Date 11/23/98	

Figure 13: Dual FPGA Top Split Power Plane Layout (fpgapwr)



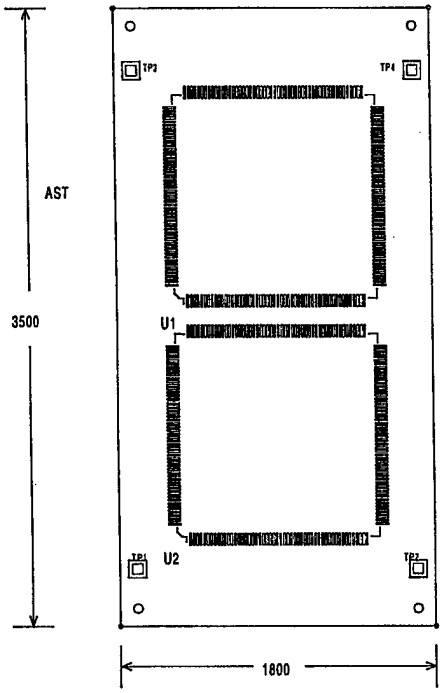
Tanner Research, Inc. 2650 E. Foothill Blvd. Pasadena, California 91107 (626)792-3000 FAX(626)432-5705	
Designer S.CHAU	Title DUAL FPGA DAUGHTER BOARD V100 Rev. A
Date 11/23/98	

Figure 14: Dual FPGA Common Ground Layout (fpgagnd)



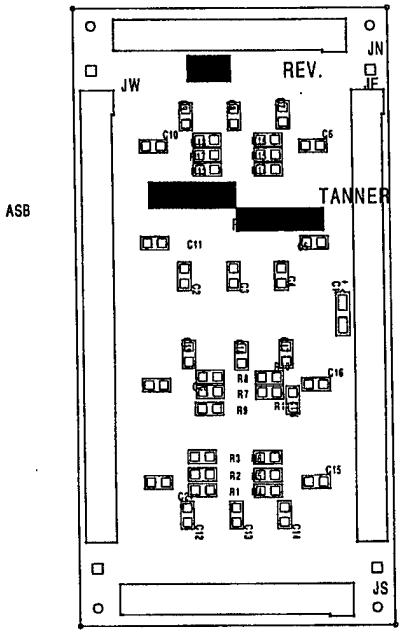
Tanner Research, Inc. 2650 E. Foothill Blvd. Pasadena, California 91107 (626)792-3000 FAX(626)432-5705	
Designer S.CHAU	Title DUAL FPGA DAUGHTER BOARD V100 Rev. A
Date 11/23/98	

Figure 15: Dual FPGA Bottom Layer Trace Layout (fpgabot)



Tanner Research, Inc. 2650 E. Foothill Blvd. Pasadena, California 91107 (626)792-3000 FAX(626)432-5705	
Designer S.CHAU	Title DUAL FPGA DAUGHTER BOARD V100 Rev. A
Date 11/23/98	

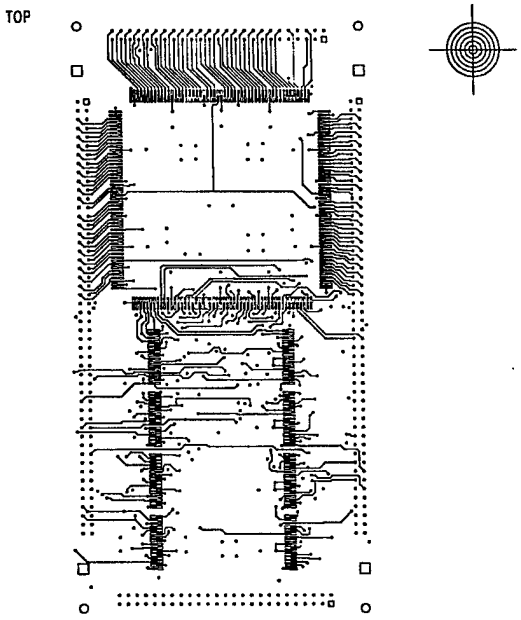
Figure 16: Dual FPGA Top Assembly Drawing (fpgaast)



Tanner Research, Inc. 2650 E. Foothill Blvd. Pasadena, California 91107 (626)792-3000 FAX(626)432-5705	
Designer S.CHAU	Title DUAL FPGA DAUGHTER BOARD V100 Rev. A
Date 11/23/98	

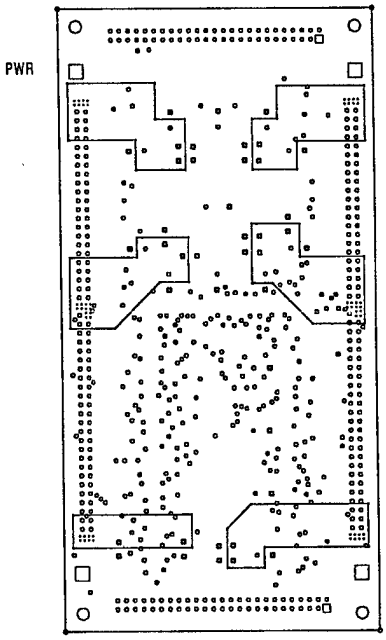
Figure 17: Dual FPGA Bottom Assembly Drawing (fpgaasb)

7 Appendix C: FPGA plus RAM Module Wiring



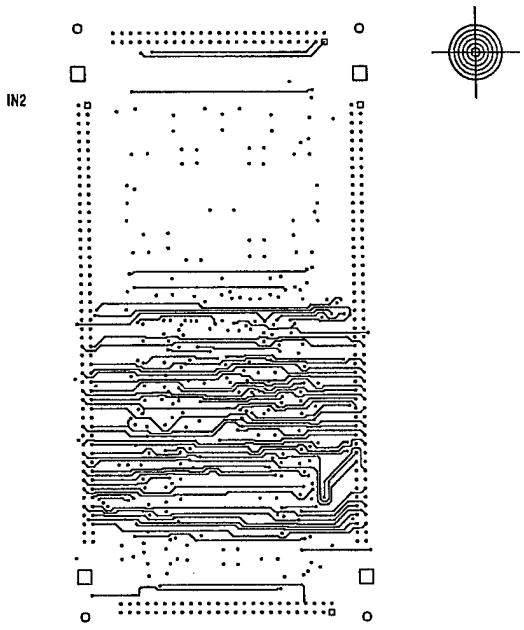
Tanner Research, Inc.	
2650 E. Foothill Blvd. Pasadena, California 91107 (626)792-3000 FAX(626)432-5705	
Designer S. CHAU	Title FPGA / RAM DAUGHTER BOARD V100 REV. A
Date 11/25/97	

Figure 18: FPGA Plus Ram Top Layer Trace Layout (fprmtop)



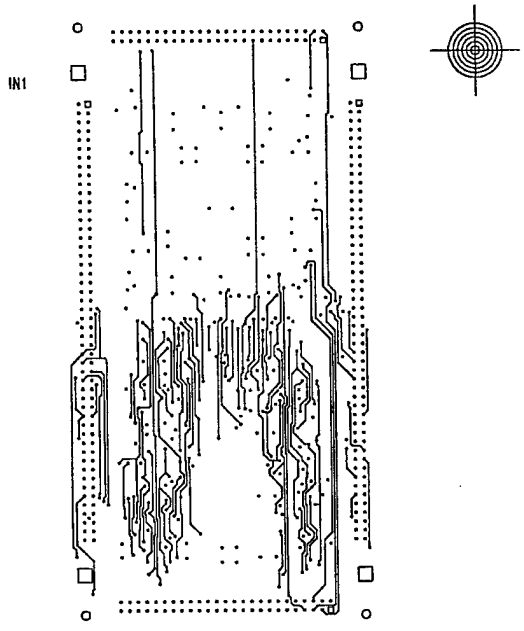
Tanner Research, Inc.	
2650 E. Foothill Blvd. Pasadena, California 91107 (626)792-3000 FAX(626)432-5705	
Designer S. CHAU	Title FPGA / RAM DAUGHTER BOARD V100 REV. A
Date 11/25/97	

Figure 19: FPGA Plus Ram Split Power Plane Layout (fprmpwr)



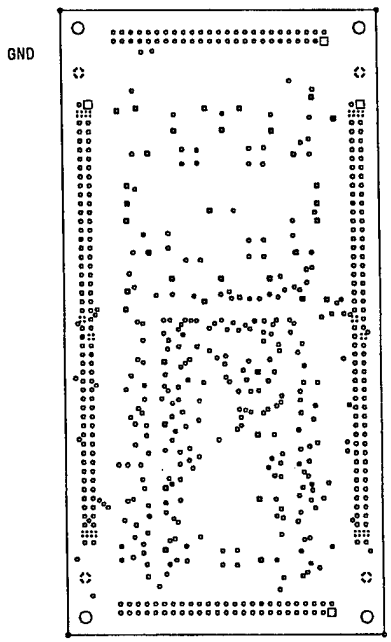
Tanner Research, Inc. 2650 E. Foothill Blvd. Pasadena, California 91107 (626)792-3000 FAX(626)432-5705	
Designer S. CHAU	Title FPGA / RAM DAUGHTER BOARD V100 REV. A
Date 11/25/97	

Figure 20: FPGA Plus Ram 2nd Internal Trace Layout (fprmin2)



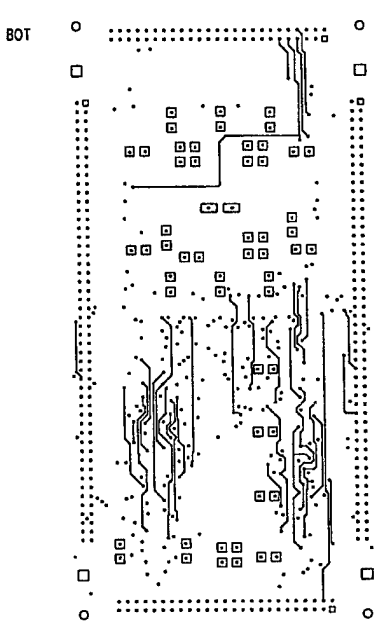
Tanner Research, Inc. 2650 E. Foothill Blvd. Pasadena, California 91107 (626)792-3000 FAX(626)432-5705	
Designer S. CHAU	Title FPGA / RAM DAUGHTER BOARD V100 REV. A
Date 11/25/97	

Figure 21: FPGA Plus Ram 1st Internal Trace Layout (fprmin1)



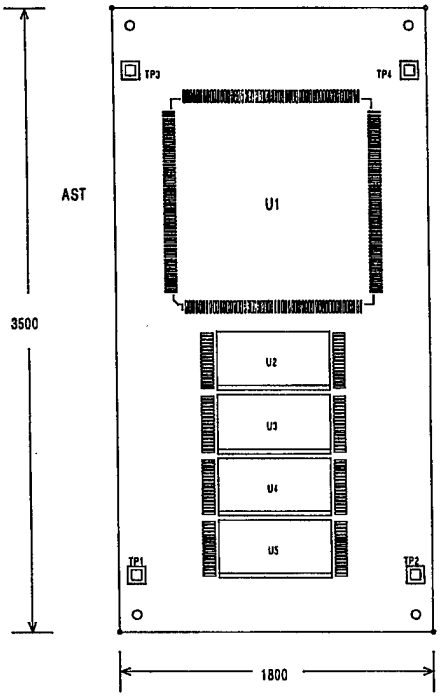
Tanner Research, Inc. 2650 E. Foothill Blvd. Pasadena, California 91107 (626)792-3000 FAX(626)432-5705	
Designer S. CHAU	Title FPGA / RAM DAUGHTER BOARD V100 REV. A
Date 11/25/97	

Figure 22: FPGA Plus Ram Common Ground Plane Layout (fprmgnd)



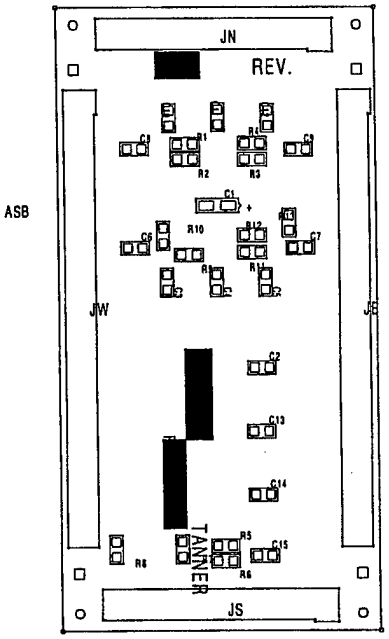
Tanner Research, Inc. 2650 E. Foothill Blvd. Pasadena, California 91107 (626)792-3000 FAX(626)432-5705	
Designer S. CHAU	Title FPGA / RAM DAUGHTER BOARD V100 REV. A
Date 11/25/97	

Figure 23: FPGA Plus Ram Bottom Layer Trace Layout (fprmbot)



Tanner Research, Inc. 2650 E. Foothill Blvd. Pasadena, California 91107 (626)792-3000 FAX(626)432-5705	
Designer S. CHAU	Title FPGA / RAM DAUGHTER BOARD V100 REV. A
Date 11/25/97	

Figure 24: FPGA Plus Ram Top Assembly Drawing (fprmast)



Tanner Research, Inc. 2650 E. Foothill Blvd. Pasadena, California 91107 (626)792-3000 FAX(626)432-5705	
Designer S. CHAU	Title FPGA / RAM DAUGHTER BOARD V100 REV. A
Date 11/25/97	

Figure 25: FPGA Plus Ram Bottom Assembly Drawing (fprmasb)

8 Appendix D: Motherboard Schematics

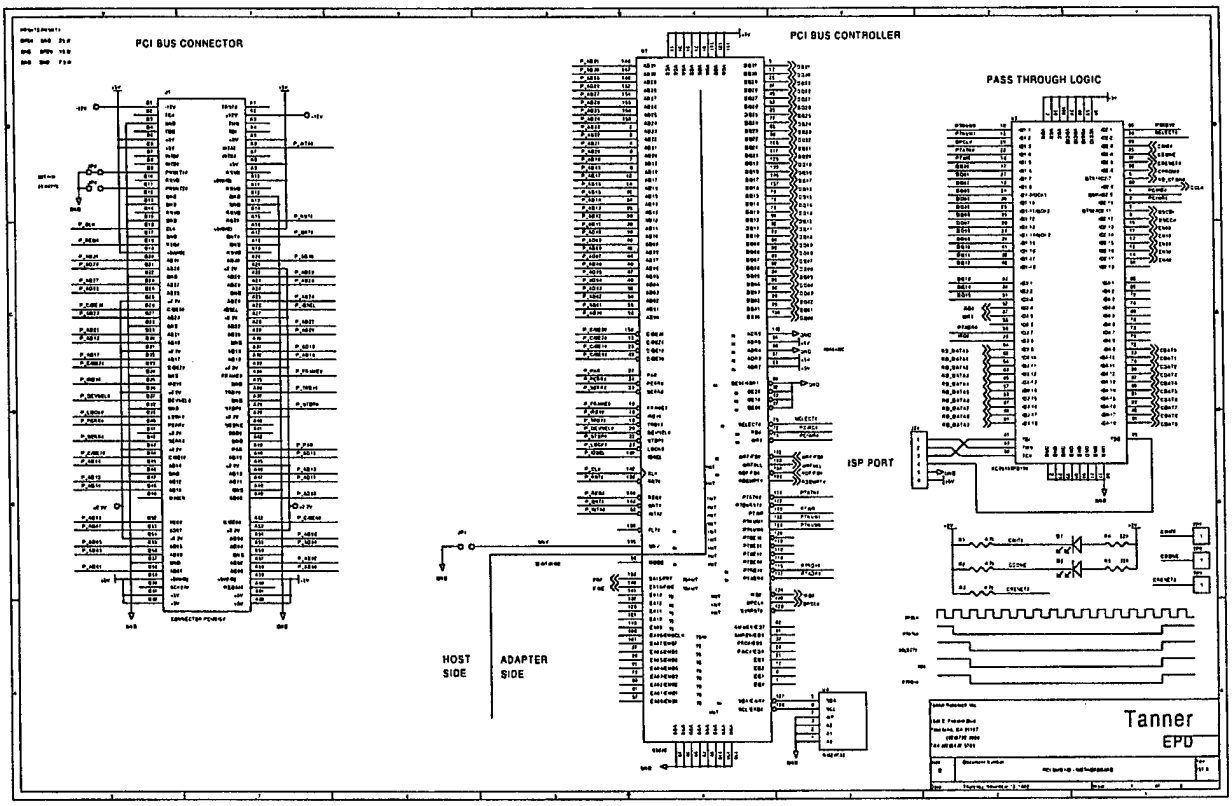


Figure 26: Motherboard PCI Bus I/O Interface

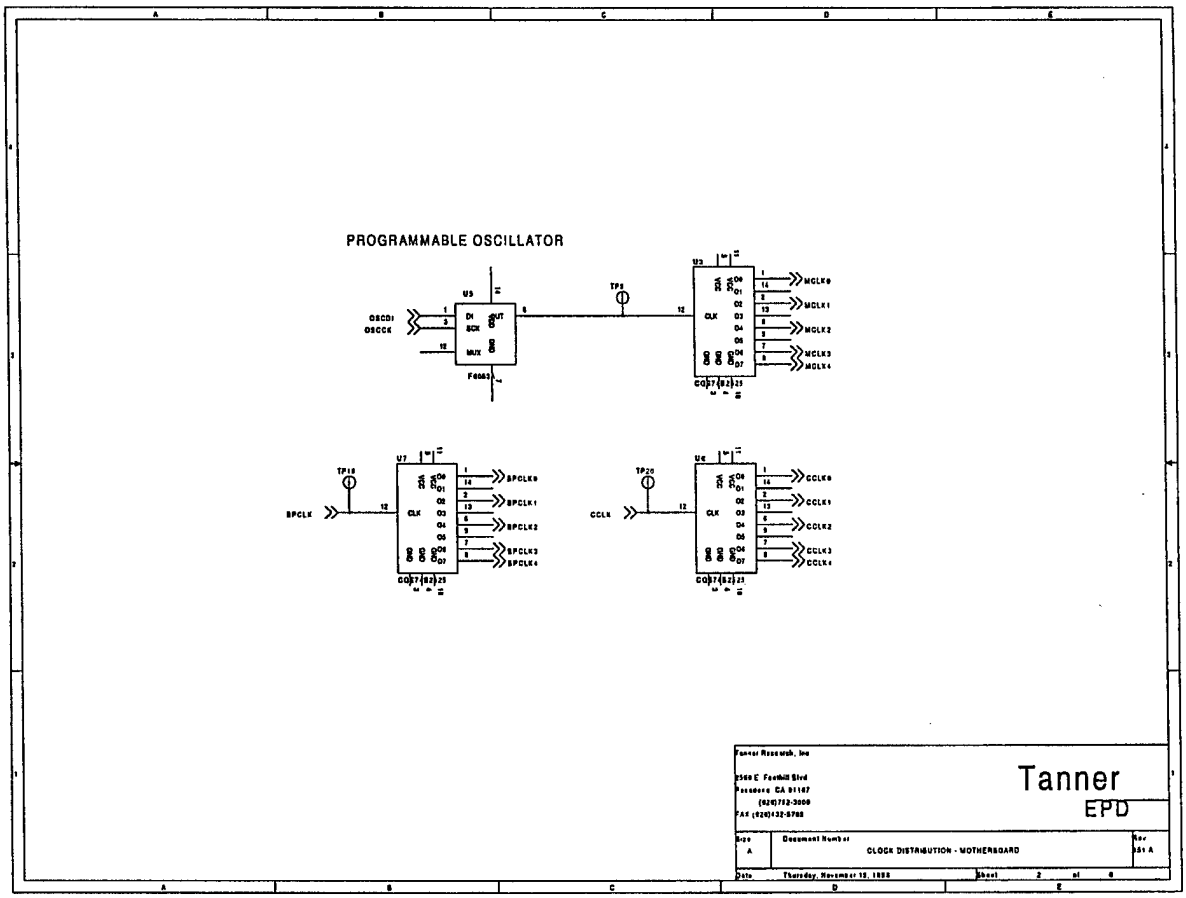


Figure 27: Motherboard Clock Buffering and Distribution

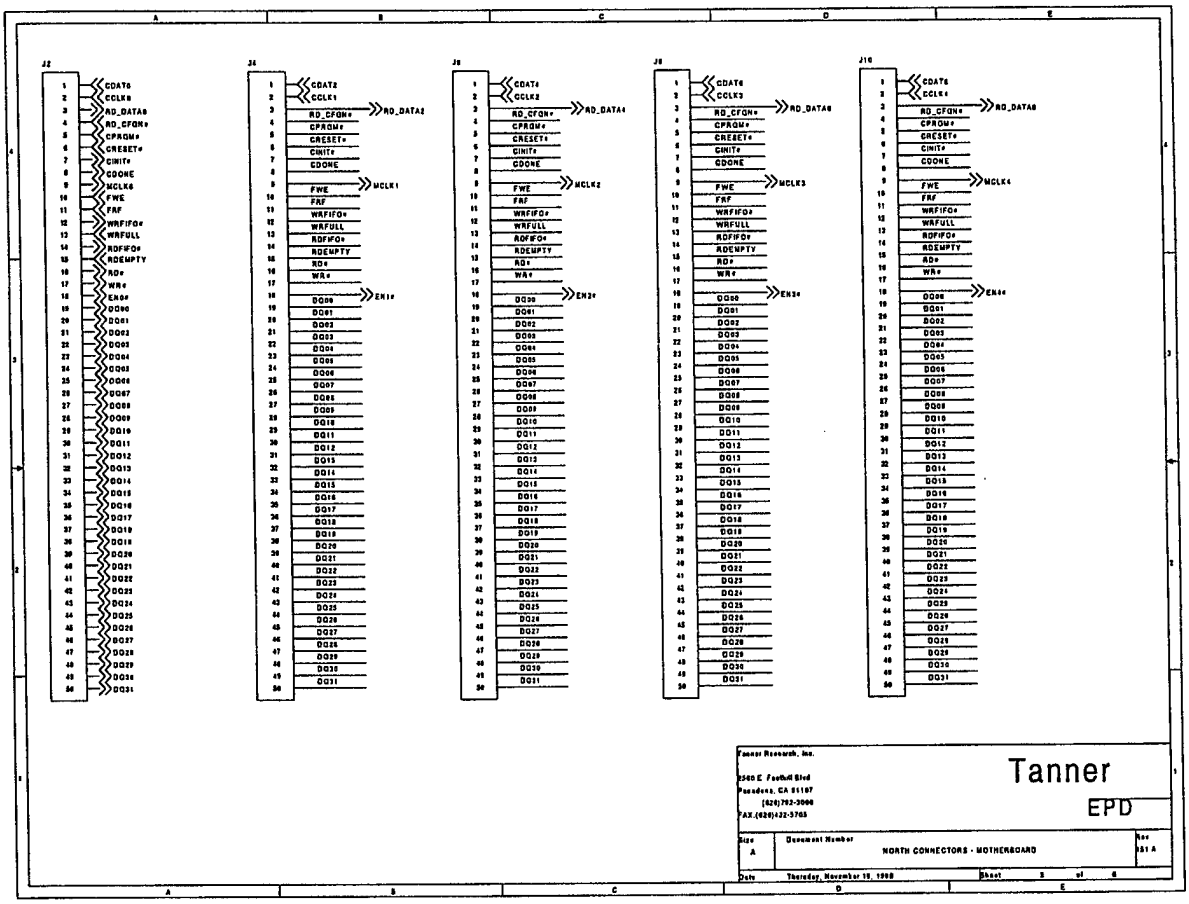


Figure 28: Motherboard North Connectors

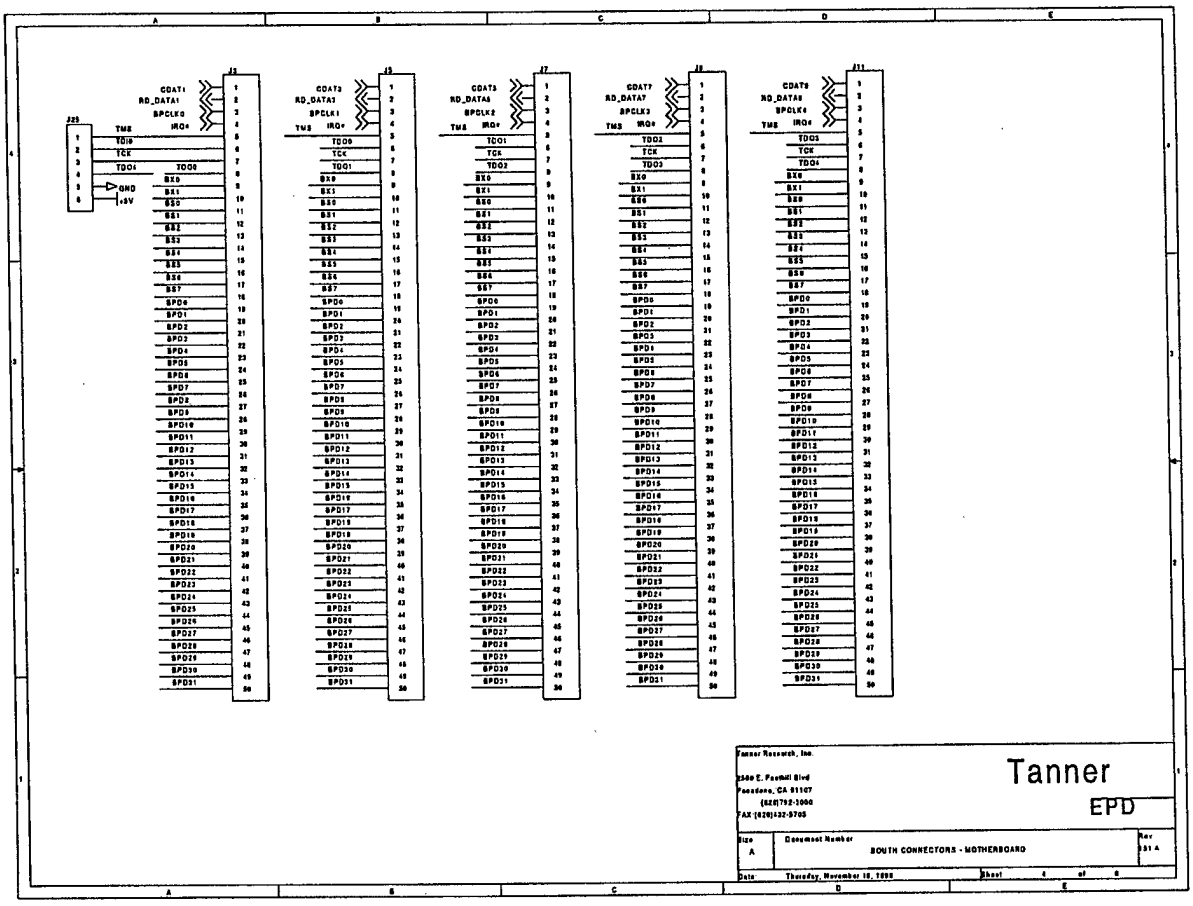


Figure 29: Motherboard South Connectors

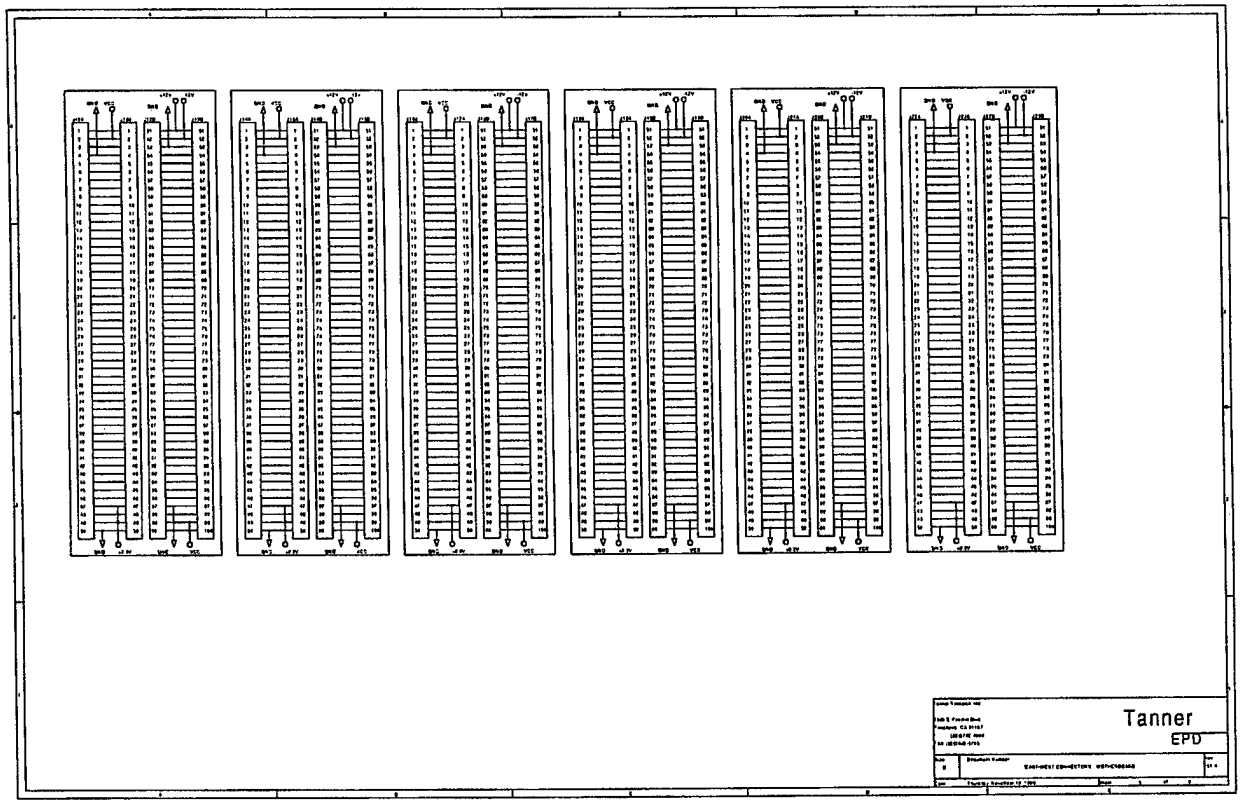


Figure 30: Motherboard East / West Connectors

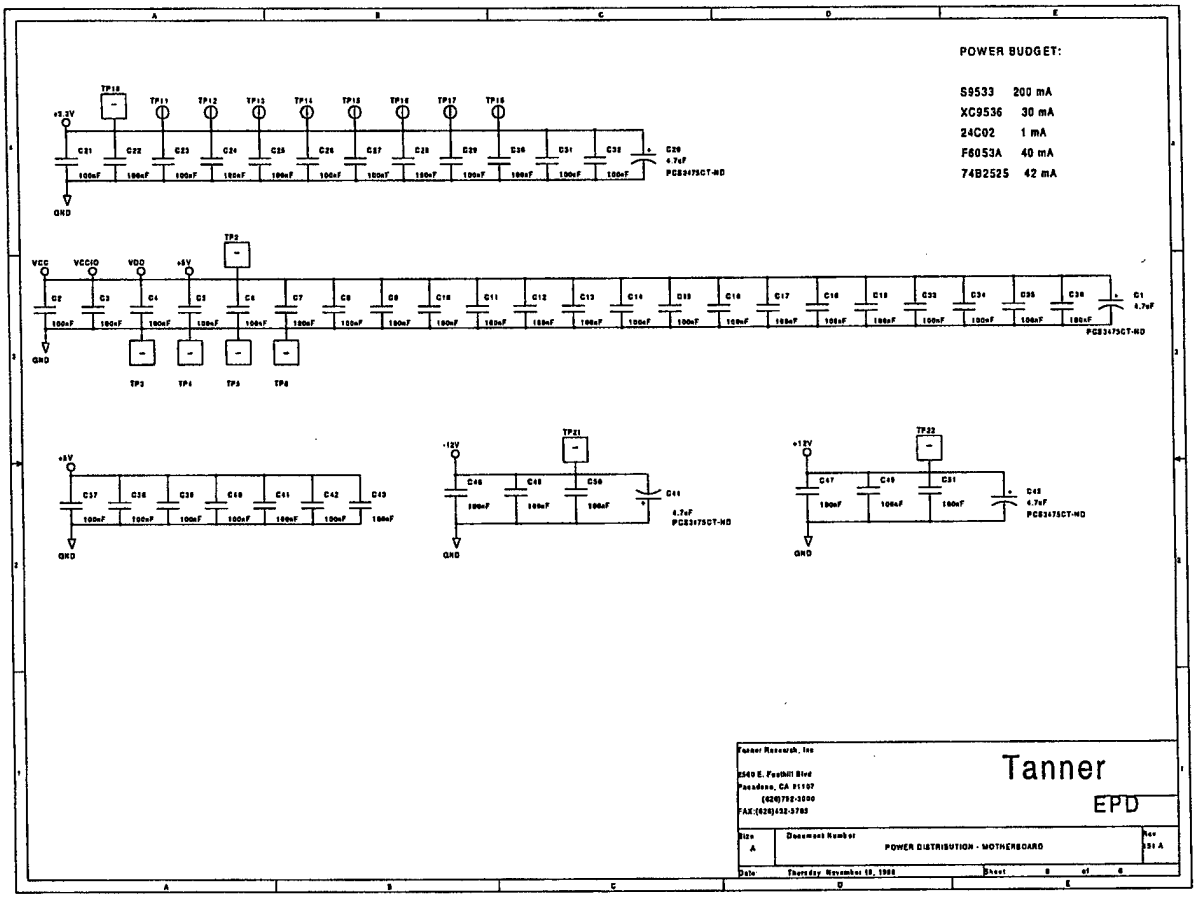


Figure 31: Motherboard Power Decoupling and Distribution

9 Appendix E: Dual FPGA Module Schematics

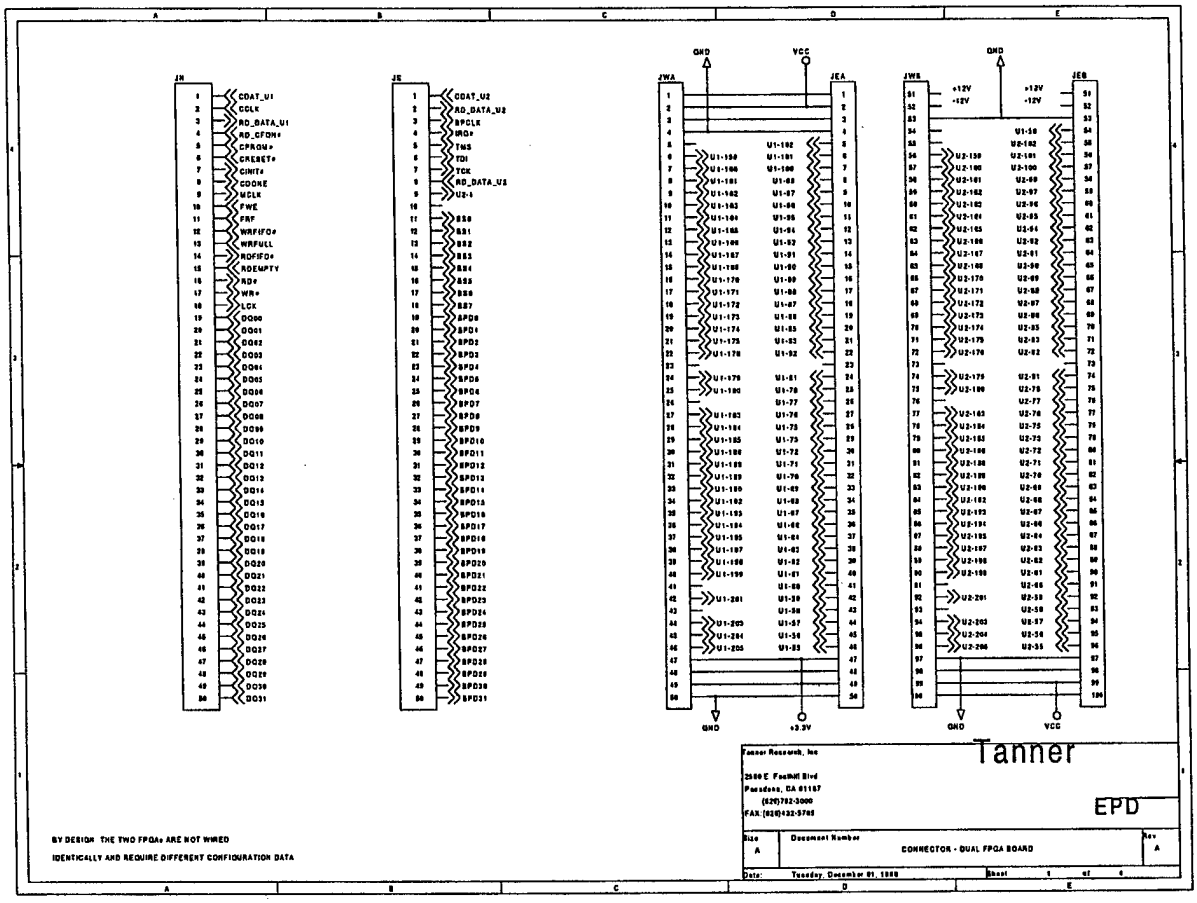


Figure 32: Dual FPGA Module Interface Connectors

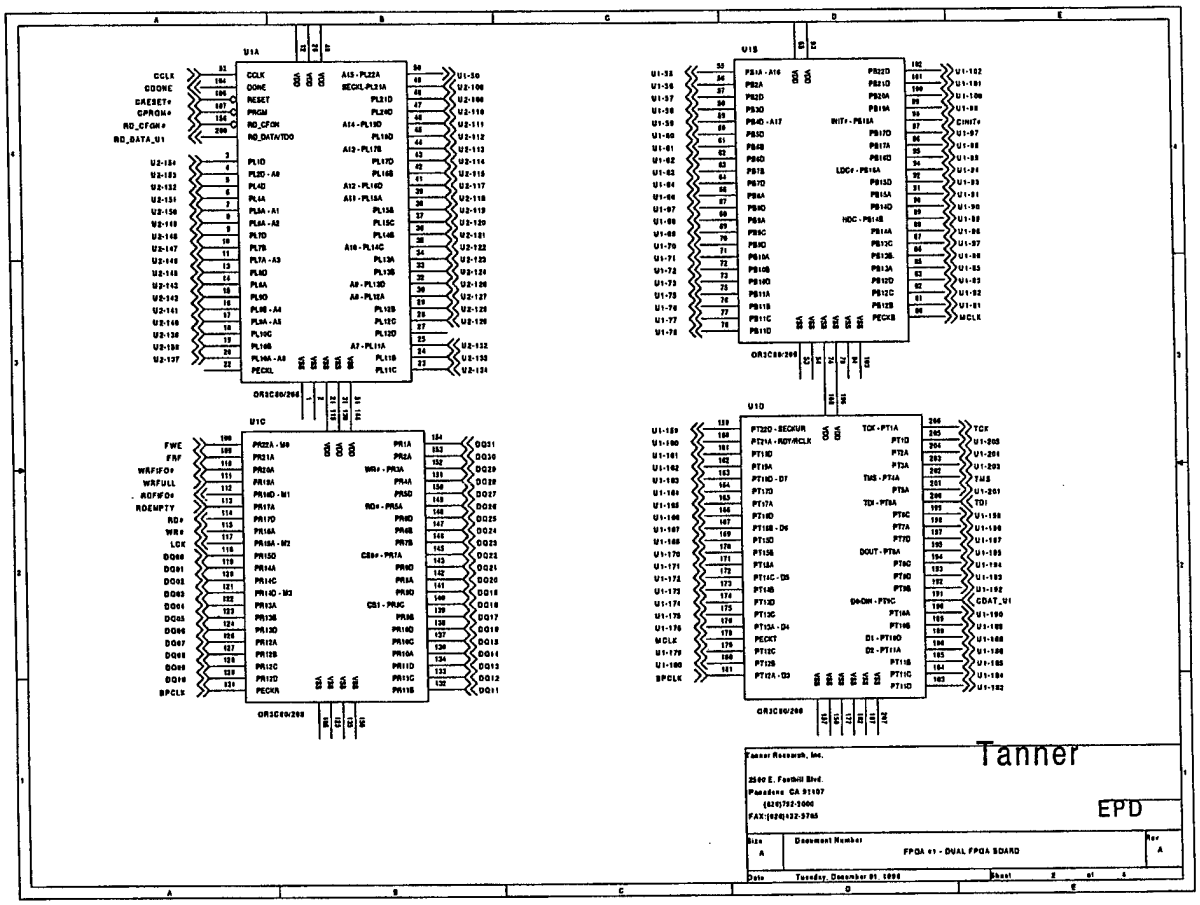


Figure 33: Dual FPGA Module: 1st FPGA Shown as Four Quadrants

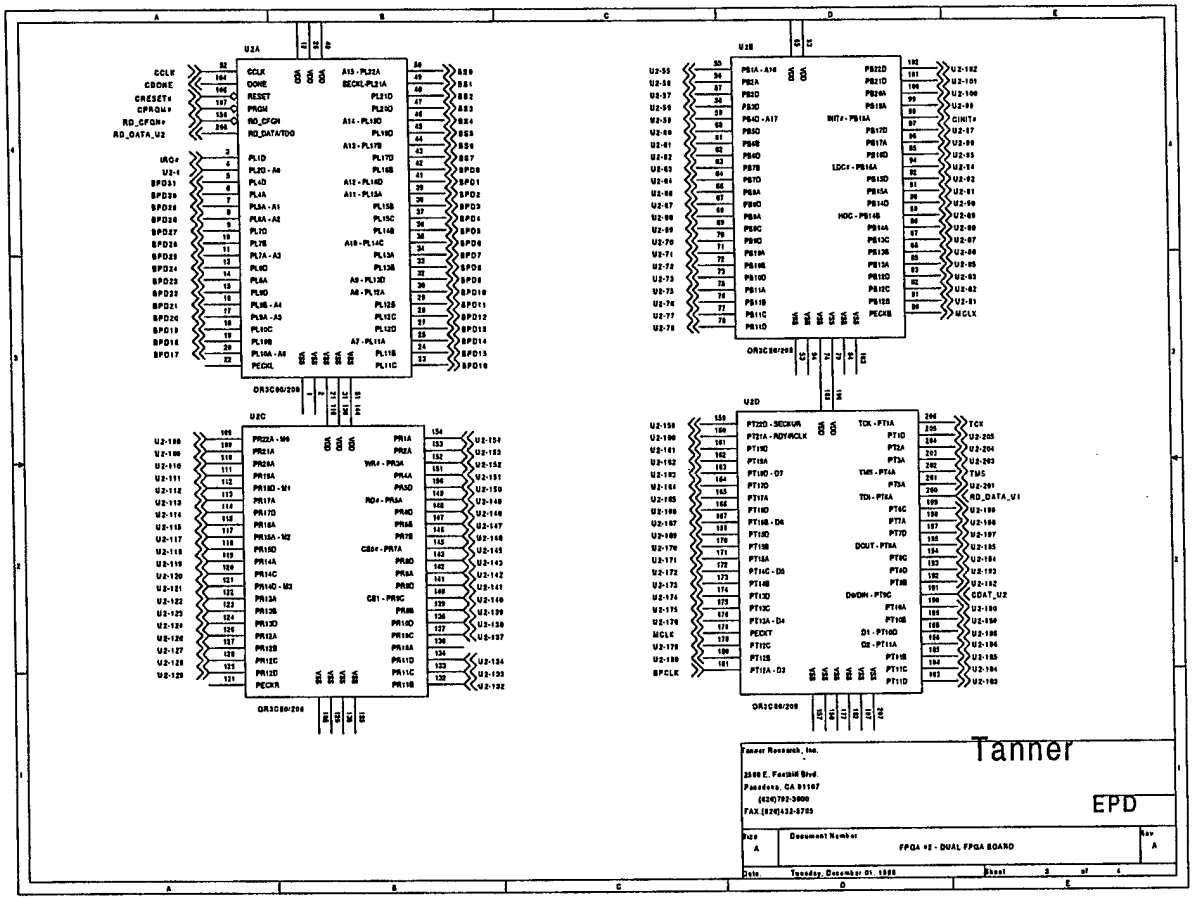


Figure 34: Dual FPGA Module: 2nd FPGA Shown as Four Quadrants

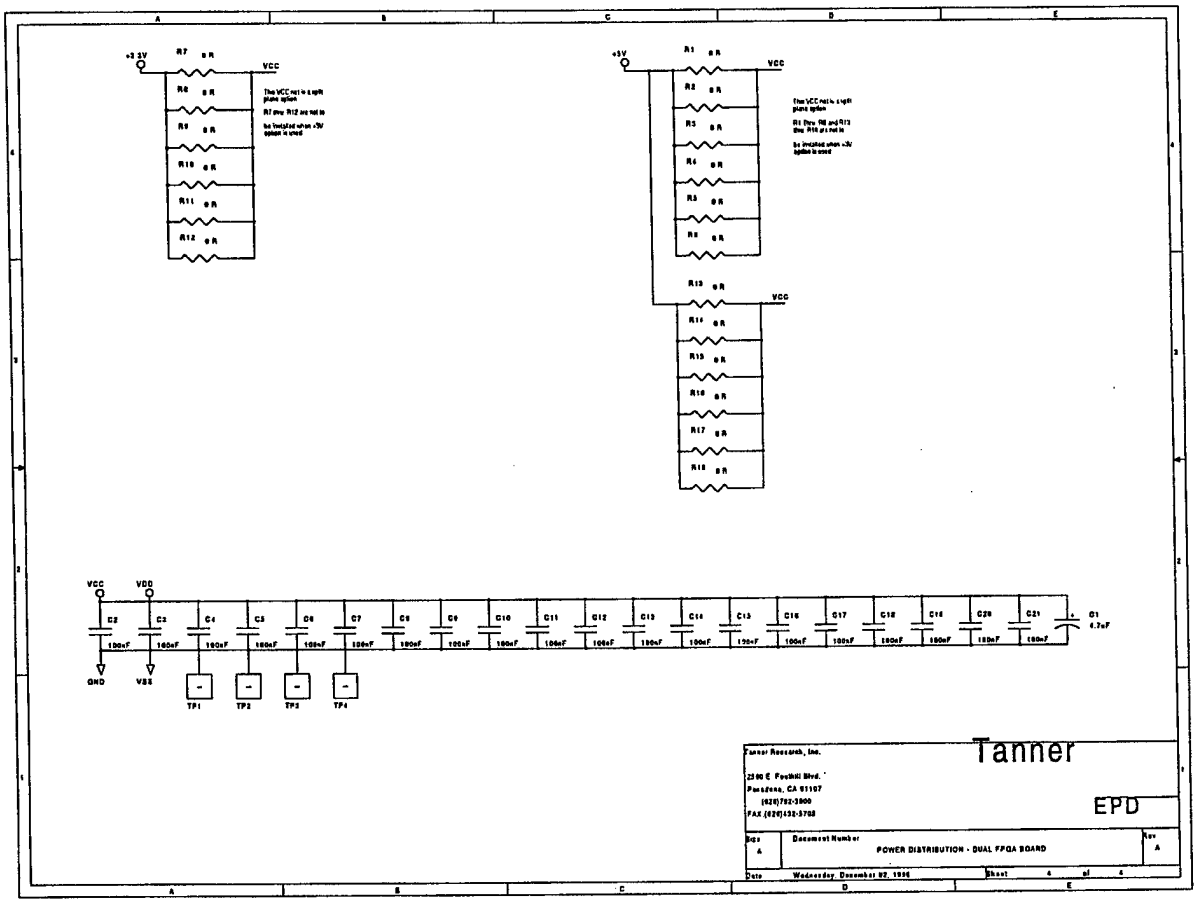


Figure 35: Dual FPGA Module: Power Decoupling and Distribution

10 Appendix F: FPGA plus RAM Module Schematics

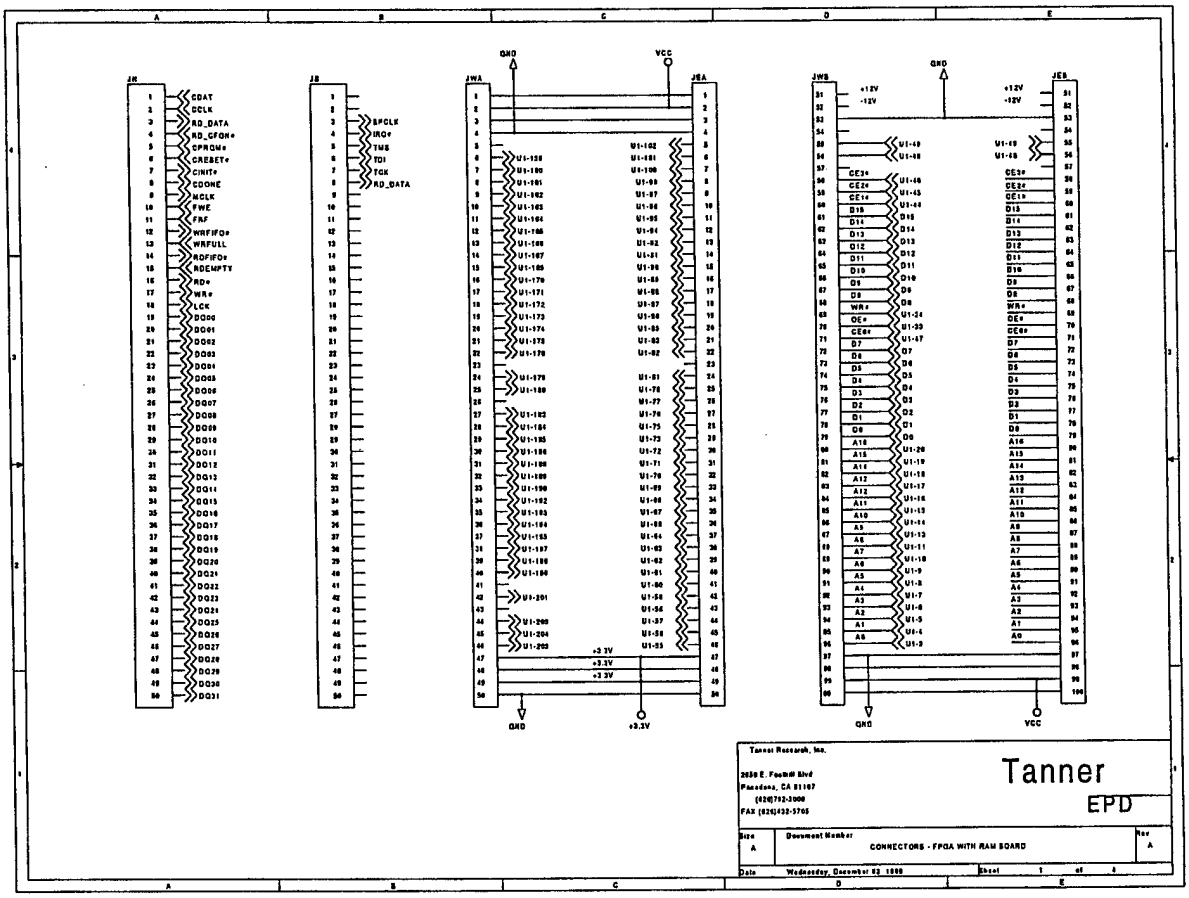


Figure 36: FPGA plus RAM Module Interface Connectors

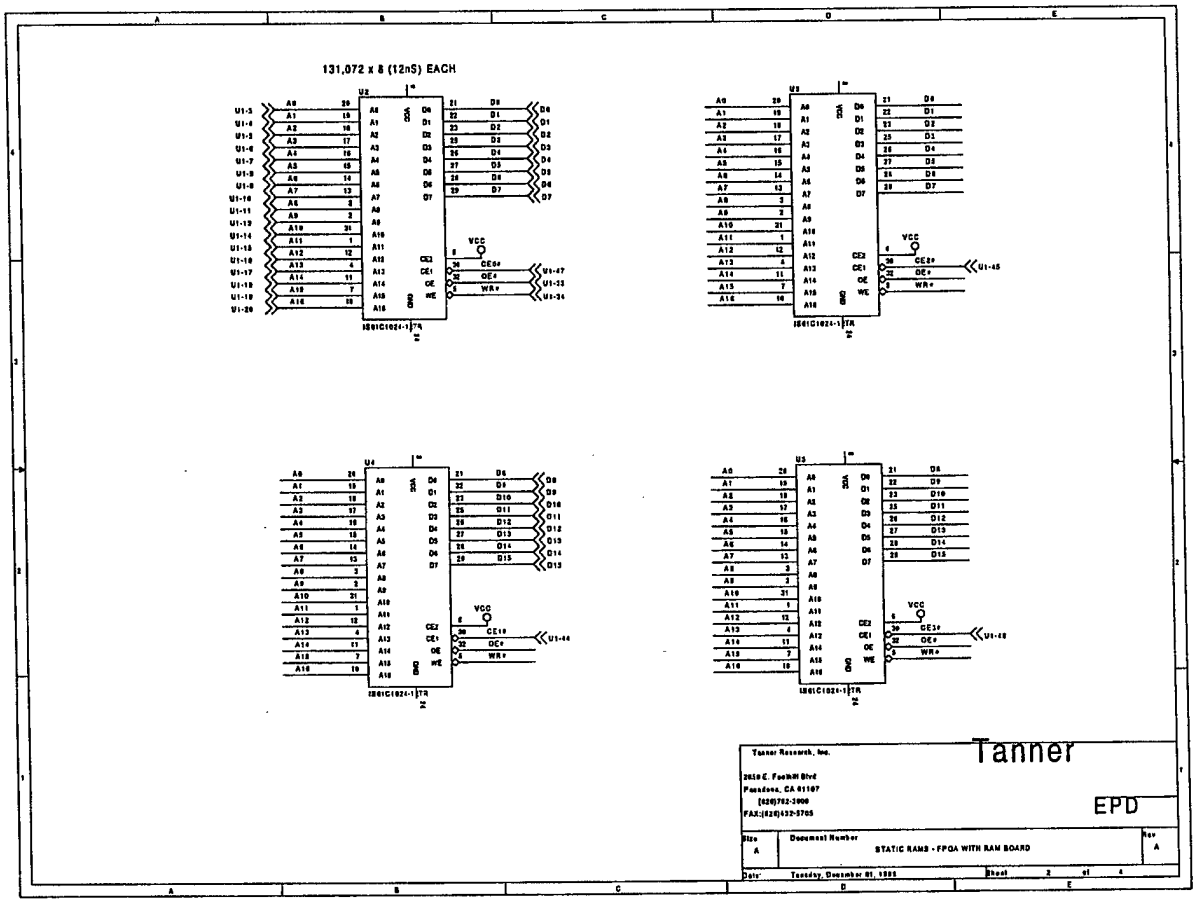


Figure 37: FPGA plus RAM Module: Static RAMs

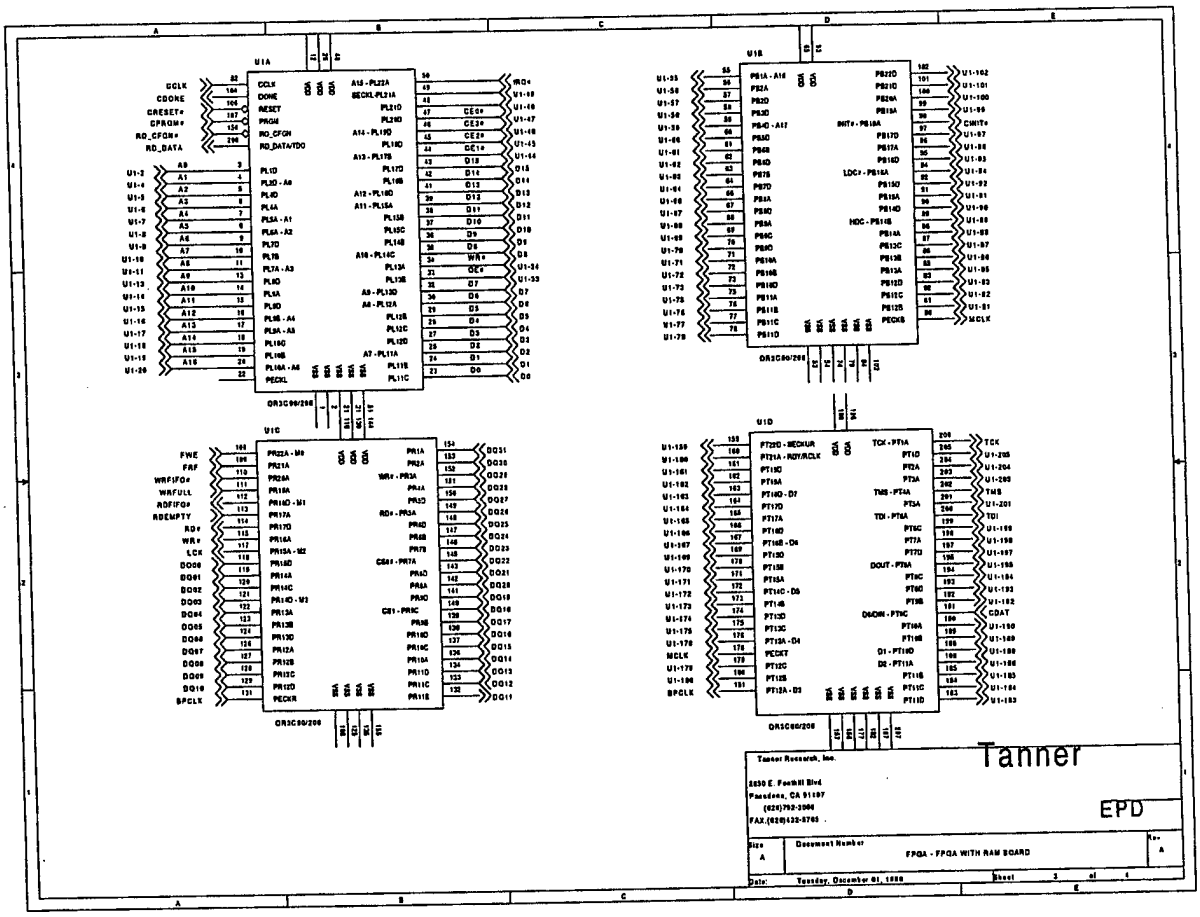


Figure 38: FPGA plus RAM Module: Single FPGA Shown as Four Quadrants

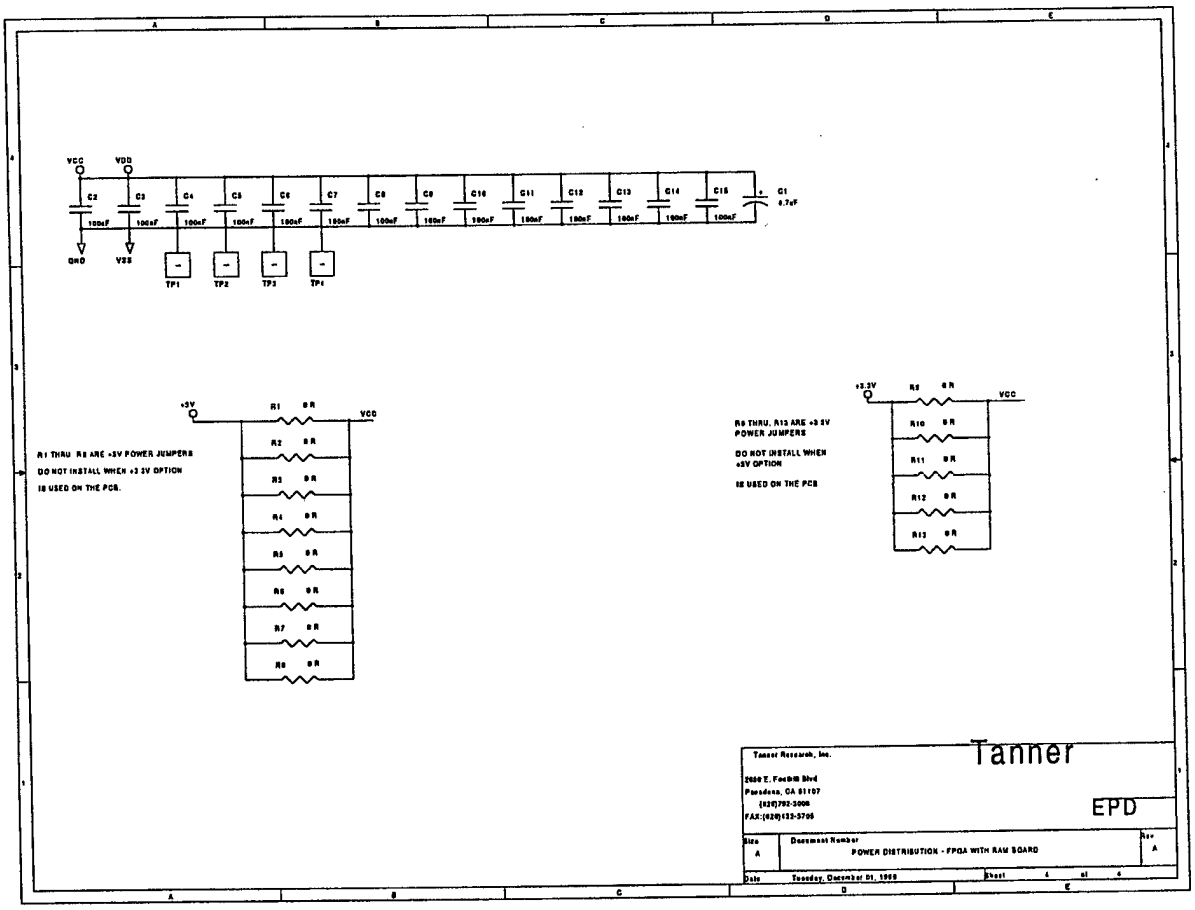


Figure 39: FPGA plus RAM Module: Power Decoupling and Distribution